DEVELOPING THREE-DIMENSIONAL (3D) TOPOLOGICAL MODEL FOR 3D GIS

Alias Abdul Rahman

Research Project (IRPA) Vot # 74152
Project # 09-02-060097 EA001

Faculty of Geoinformation Science and Engineering
Universiti Teknologi Malaysia

DECEMBER 2005

ABSTRACT

3D spatial data modeling is one of the key research problems in 3D GIS. More and more applications depend on these 3D spatial data. Mostly, these data are stored in Geo-DBMSs. However, recent Geo-DBMSs do not support 3D primitives modeling, it only able to describe a single-attribute of the third-dimension, i.e. modeling 2.5D datasets that used 2D primitives (plus a single z-coordinate) such as polygons in 3D space. This research focuses on 3D topological model based on space partition for 3D GIS, for instance, 3D polygons or tetrahedron form a solid3D object. Firstly, this report discusses formal definitions of 3D spatial objects, and then all the properties of each object primitives will be elaborated in detailed. The author also discusses methods for constructing the topological properties to support object semantics is introduced. The formal framework to describe the spatial model, database using Oracle Spatial is also given in this report. All related topological structures that forms the object features are discussed in detail. All related features are tested using real 3D spatial dataset of 3D building. Finally, the report concludes the experiment via visualization of using AutoDesk Map 3D.

ABSTRAK

Permodelan data spatial 3D merupakan salah satu masalah dalam GIS 3D. Banyak aplikasi GIS dan gunapakai GIS semakin memerlukan kepada data spatial 3D dan kebanyakan data seperti ini disimpan dalam Geo-DBMS. Tetapi kebanyakan Geo-DBMS ini tidak menyokong primitif data spatial 3D. Penyelidikan ini menfokus kepada pembangunan topologi yang menyokong data spatial 3D dan seterusnya digunapakai dalam permasalahan analisis 3D bagi data spatial 3D. Untuk ini, beberapa tool sedia ada telah digunakan, seperti Autodesk 3D Map dan Oracle Spatial. Projek ini juga telah menerbitkan satu model data spatial 3D, iaitu Condedensed Spatial Model (CoS). Model ini berguna untuk mendapatkan topologi 3D bagi data spatial 3D.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AD	Attribute Data
ADT	Abstract Data Type
DBMS	Database Management System
ERM	Entity Relationship Model
ESTDM	Event-based Spatio-temporal Data Model
GF	Geographical Feature
GFS	Geographical Feature State
GIS	Geographical Information Systems
GP	Geographical Phenomena
GPS	Global Positioning System
HIS	Hydrological Information System
ITS	Intelligent Transport System
MHIS	Malaysian Hydrological Information System
ODMG	Object Database Management Group
OMT	Object Management Techniques
OODBMS	Object-oriented Database Management System
ORDBMS	Object-Relational Database Management System
RDBMS	Relational Database Management Systems
SAR	Synthesized Aperture Radar
SD	Spatial Data
SDT	Spatial Data Types
SQL	Structured Query Language
STD	Spatio-Temporal Databases
STGIS	Spatio-temporal Geographical Information Systems
STORM	Spatio-temporal Object-relation Model
TD	Temporal Data
TE	Transaction Ending time

TS	Transaction Starting time
VE	Valid Ending time
VS	Valid Starting time

CHAPTER 1

INTRODUCTION

1 Introduction

Currently, a very strong development in the field of software applications is moving towards web-enabled systems. The reason for this is because information technology and its infrastructure are offering more and more possibilities to share data, applications and their logic within networking environments. Here, the most obvious example is the success of the Internet as a platform for communication in all kind of variants. Within the last years, it has been proven that Internet applications are working effectively and everybody is able to benefit from it. Among many others, the success of e-commerce is reflecting this story of success. However, not only the Internet is considered as “web-enabled”. For example, there are a large number of web-applications within local area networks (LAN), like an enterprise communication platform. In the context of web-enabled applications, the science of information technology deal with distributed computing.

The field of geo-informatics profits by adopting these developments as well. The advantages are obvious, web-enabled applications are saving resources and making geo information accessible to everyone who can benefit from it. The increasing number and the popularity of Internet applications related to geo-information are reflecting this evolution. Favored services are including for instance interactive city guides, location finders or online-routing. Furthermore, great efforts among the “big players” of the GIS community towards web-orientated GIS are reported on several newsletters and magazines monthly. Such systems are called web-enabled- or Internet GIS. Recently, the general term of “Distributed GIS” is becoming more and more popular as well. Here, the OpenGIS Consortium (OGC) plays an important role. As an organization of many companies and institutes, the OGC is standardizing data sources and interfaces for geo-related Web Services – in this context called “GeoServices” (ESRI, 2001). Presently, there are already GeoServices available and accessible. However, most of the current web-orientated applications are mainly restricted to visualization (Peng and Tsou, 2003). Less comprehensive GeoServices are available for the second dimension - three-dimensional (3D) GIS is limited to visualization only. Systems which are offering core 3D functionality including data management, -edition, -analysis and interactive user interfaces are only available for single desktop use.

1.2 Problem Background

In terms of possible applications, especially managers are able to benefit from 3D GIS as a base for their work. Their duty is to cover the whole process of planning complex city centers, for instance construction simulation, emission/waste control, disaster management or telecommunication/energy supply. Beside, tourism and facility management are able to benefit from virtual worlds as well (Altmaier and Kolbe, 2003). Laurini points out, that different authorities can benefit from information systems for urban planning. Furthermore, the public participation in decisions around this topic can also be improved. Here, especially 3D applications are able to assist users and to simplify complicated workflows (Laurini, 2001). Therefore, the demands for web-orientated 3D information systems will be increasing in future.

As a conclusion to the discussed tendency towards distributed systems, the general research question follows:

Can the Web be a reasonable medium to host 3D GIS with its tasks for urban environment?

In order to give an appropriate answer, the research is considering the four principles of GIS. Here, the topics of 3D data management, analyzing and querying data as well as publishing/visualizing 3D maps within a web environment are considered, whereas 3D data acquisition will not be the main target of the research itself. Coors (2003) points out that the processing of 3D geo-data within GIS still has to improve (Coors, 2003). **Figure 1** is showing the four main stages within GIS in conjunction to their present availability for the third dimension.

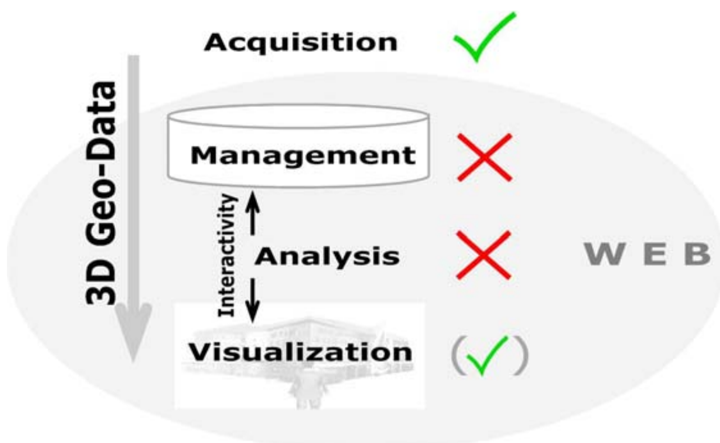


Figure 1: The 3D Geo-Data Workflow within GIS

As shown in **Figure 1**, the data management respectively the storage is most important for further processes like 3D analysis and 3D visualization as well as for providing interactivity. In fact, it is regarded as the base of any further step. In contrast to a 2D environment, the third dimension is requesting different aspects. One important topic which can not be the same is the underlying data management. Previous and current research is concluding that the underlying spatial data for web-enabled 3D GIS is by far more complex than 2D geo-information (Coors, 2003). For this reason different characteristics in terms of data modeling and –management in order to integrate 3D GIS analysis and to visualize results have to be addressed.

Can the Web be a medium to manage 3D spatial data in databases?

Fundamental research in the field of 3D spatial data modeling for urban environments has been done mainly by Molenaar (1990), Pigot (1992), Tempfli and Pilouk (1994), Pilouk (1996), Zlatanova (2000), Pfund (2001), Shi *et al* (2003) and Coors (2003). The specific task of web-orientated systems has been covered by Zlatanova (2000), Coors (2003) and Vries and Stoter (2003) before. However, more web-specific research has been demanded as a conclusion.

Although the most recent efforts are considering 3D visualization on the Web, applications which are dynamically serving interactive 3D geo-worlds are not fully available yet. Functionality like 3D data analysis - on top of a reasonable data management - in terms of GeoServices is still missing (Coors, 2003). Therefore, another research question will be:

Is it possible to realize GI-Services for complex 3D spatial data tasks?

Previous studies on the dynamic creation of interactive 3D worlds from spatial databases have been done by Kofler (1998), Zlatanova (2000), Coors (2003) and de Vries and Stoter (2003). Their prototype systems have shown solutions which have been integrated into web architectures. Standalone as well as browser-based user agents were used to create intuitive graphical user interfaces which are able to interact with 3D geo-information. Here the main question will be:

Until now, common results of “so-called” 3D applications are static 3D views in form of images, 3D animations like fly-through movies or interactive 3D scenes using techniques like the Virtual Reality Modeling Language (VRML) or Java3D. Whereas the first two cases are rendered “offline” (rendered to images/movies before viewed), the later one is rendered in real-time and offers 3D user-interaction. There is no doubt that up-to-date computers offer a reasonable environment to reconstruct and visualize virtual worlds very close to reality. However, the following questions are rising:

Are current web-visualization techniques able to visualize 3D spatial data in real-time?
Are there reasonable interfaces to access and interact with the data?

The research will give a scientific view about 3D GIS. Therefore, the right answers will show the research results. In addition, the summary and conclusion will face towards further discussion around the topic of web-enabled 3D GIS.
Therefore, useful existing research and approaches has to be evaluated first. Here, web integrity of 3D geo-information and its visualization are among the most critical topics. More precisely, 3D spatial data and its complex structure as well as its huge amount of information have to be considered since web-orientated applications have some limitations in terms of performance – especially band-width. Based on a reasonable data management concepts of real 3D analysis functionality and 3D visualization methods will be discussed. Regarding the data management, there is still a lack of proven concepts. Whereas the storage of 3D geometry can be realized easily, 3D topological modeling is most critical. It is among the most problematic aspects how to model complex buildings and transform their geometry into databases. Therefore, one research question is “what are the existing 3D topological models which can be used for 3D GIS in Web environments?”

Furthermore, the issue around storing corresponding attributes and textures has to be addressed. Besides, a concept for implementing spatial operators has to be constructed. Here, the question is if there are any spatial operators for 3D GIS in Web environment. In addition, the question whether methods of 3D spatial analysis have to be implemented on database or application level is an issue. In order to access, query and visualize 3D data, the integration of user agents to do so has to be evaluated. Here, the tasks on an interactive 3D GUI which is able to be integrated in a web environment are very high. It has to offer interactive 3D visualization as well as possibilities to query the database and data editing. Because geo-information and its data attributes is sensitive as well as expensive and under complicated copyright laws, issues around security and privacy of distributed GIS have to be addressed as well.

In order to prove web-integrity of 3D geo-information, this research will include an implementation which is showing the proposed concepts of 3D data management, 3D analysis and 3D visualization using state-of-the-art techniques. The solution should be harmonized with common standards like the OGC or World Wide Web Consortium (W3C) recommendations. Furthermore, the implementation should show if recent web-technologies are suitable for web-orientated 3D GIS. In order to introduce practical applications specific use cases for solving problems in the field of urban environment will be set up.

1.3 Problem Statement

The aim of this research is to design and develop a 3D spatial model and also discover current issues regarding web-orientated 3D GIS and to develop concepts to solve untouched problems on 3D web-based GIS.

The following research questions are addressed to solve the problem:

a) How to properly manage the 3D spatial data together into a database?

b) How to manage 3D spatial data relationships?

c) How effectively Spatio-temporal queries can be handled by a model using object-relational approach.

1.4 Goal of the Study

The goal of this research is the design and development of a data model for management of 3D spatial data and their relationships on a 3D web-based environment.

1.5 Objective

Objectives of this research can be divided into the main intentions which should be achieved through short-term goals. The research objectives are to:

1- Investigate current and existing concepts and approaches for web-enabled 3D GIS.

2- Design and develop a practical spatial data model for web-orientated 3D geo information in urban environments and setup a reasonable system architecture for a 3D GIS.

3- Develop and implement a prototype system including necessary web-based 3D GI-Services based on the developed spatial data model and the constructed system architecture.

4- Evaluate and test the designed and implemented prototype in a web environment.

1.6 Scope of Research

Overall, this research concentrates on web-enabled 3D GIS. Because this topic is too general and broad, the research concentrates on specific topics. The following limitations and requirements outline the research in detail.

a) Reviews and comparisons of the existing data models in the area of 3D spatial data modeling and other related areas of geographical information system.

b) Design and develop the proposed 3D spatial data model and their relationships.

c) Define a set of 3D Spatial queries to evaluate the proposed 3D Spatial data model with the specified parameters based on the handling of 3D spatial data, effectiveness in terms of space, less data redundancy and consistency.

d) Analysis and evaluation of the results of 3D Spatial queries after applying them on the proposed 3D spatial data model.

1.7 Thesis Contribution

In this thesis, data model is designed for 3D geographical information system, which can handle the 3D spatial data and its relationships. It provides an applicable platform for the efficient synthesis of 3D spatial data and related queries.

In General, the major contribution described in this thesis can be summarized as follows:

a) Design of general structure for modeling 3D spatial data in 3D geographical information system.

b) Defining the relationship rules for 3D spatial data such and also their applications on 3D GIS.

c) Efficient query handling for Spatio-temporal data by using the measures such as how better model support the analysis of spatio-temporal data, efficiency in terms of time, less data redundancy and consistency.

d) Minimize the data redundancy and data inconsistency in the 3D spatial data model.

1.8 Thesis Organization

The aim of this proposal is to introduce the research subject in detail. The thesis is divided into the following chapters.

Chapter 2 is based on the core literature review. The second chapter gives detailed information about the background and the state-of-the-art of web-orientated 3D GIS. Here, fundamental information about developments in the field of Information Technology and about 3D topology will be covered. Furthermore, recent research around the topic of 3D data management as well as 3D modeling of spatial data will be discussed. Here, already available approaches will be discussed.

Chapter 3 – Methodology - the research methodology - proposes how the goals will be achieved. Therefore, the four phases of literature review, conceptual design of the system, implementation as well as testing and evaluating the system are differentiated. The last Chapter of this proposal sets expected results of the project. The appendix is including the research schedule and a draft of the thesis structure.

Next, Chapter 4 is the design and architecture of the developed system. In this Chapter we have explained our designed 3D topological model along with the architecture of the system developed in this research.

Afterwards - in Chapter 5 – This chapter describes the results and experiments performed for this research to by implementing proposed methodology to get the desired results.

Chapter 6 – This chapter concludes our research performed on 3D topological modeling for 3D GIS.

CHAPTER 2

LITERATURE REVIEW

2.1 Background and State-of-The-Art

As stated before, the topic of web-enabled 3D GIS has already been discussed by several people. Based on the literature review, the following sections give a brief overview about the status of current activities and technologies related to the research.

2.1.1 Developments in the Field of Information Technology

Information Technology is the base field for any computer-related tasks. Therefore, it can be seen as the key which is affecting many different fields of the society. At the moment, the terms of "Internet" and "Web" are being widely used. In fact, the new technologies and its corresponding web-orientated applications are simplifying workflows everywhere. Previously, communication was mainly possible through the telephone or fax. Today, applications like e-commerce platforms have transformed these advantages into completely new fields of profitable applications. The success of this evolution has several reasons. On the one hand, there are the fast-paced developments regarding the introduction of new innovative techniques. On the other hand, one is able to recognize a general acceptance of new and useful solutions. Both sides are indicators for this huge progress and finally the success. The following paragraphs provide an overview on recent developments in the field of what one can call "The Web". They are regarded as the fundamentals for this research.

2.2 An Introduction into Distributed Computing

The increasing penetration of computers changed the way of working in many ways. One idea computerization brought up was the task of Electronic Data Interchange (EDI). In fact, the beginning of distributed computing is originated here, because the requirements of EDI were concluding in applications we are recognizing as distributed systems today. At the beginning of this "new age", the intention was only to share text data more easily through networks. Other media elements, like images, have already been served since the early nineties. Later, more functionality e.g. in terms of user-interaction was added to web sites. For example, users were already able not only to retrieve data. Furthermore, they could edit and manipulate data stored somewhere else. On the technical side, client-server architectures were the base for these operations.

Today, while the technical base can be still a client-server model, the evolution reached another level. In addition to share data, users are able to offer applications and their logic as well. Most recently, the development is more and more including a process oriented Web. In this context, we are talking about web-enabled information systems or distributed systems/computing. Distributed means, the locality of the components or objects comprising an application can be on different computers connected through a network.
Figure **2** is showing the principal architecture of distributed systems (Nagappan *et al*, 2003).

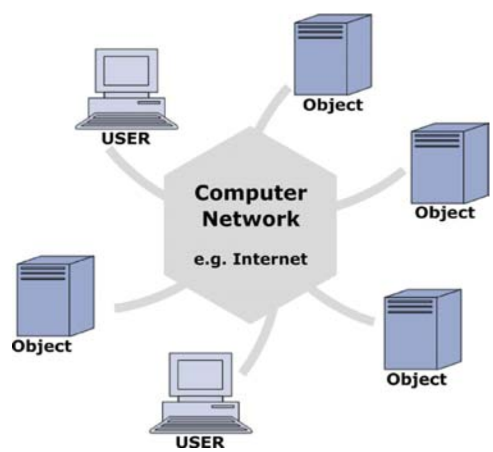


Figure 2: Distributed Computing

This very general figure is showing a distributed computing architecture with including several objects and clients. The amount of the objects and users is indefinite and therefore the value can be very high in large-scale systems. Distributed computing is inheriting the concepts of Object-Oriented Programming (OOP). Because of that, all the advantages of OOP are transformed into distributed systems as well.

In some cases, the term of distributed computing is not defined in the same way (Erlanger, 2002). It is bothering the field of grid computing as well and one can get confused about it. However, these terms in are generally different and defined as the following. In contrast to distributed computing, grid computing is facing the use of many machines – combined to a computer grid - in order to perform large-scale processing tasks. Within a computer grid, the resources of many computers are merged to a more powerful machine to run certain tasks more quickly.

Distributed computing changed many things in the field of software application development. Whereas traditional software was designed for the use on certain machines only, distributed components are more flexible and open. Beside the usual advantages of OOP, the following characteristics are showing why distributed programming became so popular and why it is unique (Nagappan *et al*, 2003):

- Distributed computing is supporting the idea of the collaboration of multiple applications. Hence, the construction of core systems using different components of other "expert" systems is possible more easily.

- The availability of services is high through clustering on multiple machines. For example if one server is failing to process the request it can be forwarded to another machine which is processing the task or hosting a copy of the service in order to process. Besides, with the extension to grid computing, the performance of large-scale processes can be increased as well. In order to do so, distributed systems are providing an environment to run them on several CPUs.

- Scalability and Extensibility: as stated before, within a distributed computing environment, there are no limits regarding the use for components. Applications can be extended easily, whereas certain objects can be reused as well. This means that distributed components of system are not only restricted to this one. Other systems are able to benefit from them as well.

- Because distributed systems are recognized as decentralized, usually massive applications are split into smaller components automatically. As a result, the development cycle time of distributed systems is lower compared to traditional mainframe applications.

- Cutting costs: once the infrastructure has been set up, there are nearly no further costs incurring. Reusing components, especially those ones which are not used at certain times, is the base for efficient use of the infrastructure.

However, distributed systems have to be deployed for the right purpose. For example, as stated before, there is no need to process small-scale tasks within distributed systems. Furthermore, if networking infrastructure is not available, distributed computing fails from the roots. Beside this, other problems are occurring while the realization of distributed computing. These are especially security and privacy questions, service payment possibilities and common standard implementations.

Typical distributed computing systems are located in the field of e-commerce. Here, for example business-to-business (B2B) solutions like online marketplaces are realized using distributed computing technologies. Other applications are Content Management Systems (CMS) of big enterprises or Customer Relationship Systems (CRS). Another big field of applications is distributed computer games, like multi-player games. In fact, nearly any industry is able to take profits out of distributed systems. The following chapters are giving technical details about the different system architectures of distributed systems. The main focus is set on the state-of-the-art technology around the XML family.

2.3 Traditional Client-Server Architectures

A very important milestone towards distributed programming was the introduction of client-server systems. Such systems were offering advanced possibilities within computer networks for the first time. Client-Server models are used until now. Whereas the client is responsible for handling the presentation and the logic of the user application, the server is organizing the application and is hosting or accessing the data. Data source can be hosted on the same machine or retrieved from another component In terms of physical abstraction a client-server system is containing at least two-layers - known as two-tier systems as well. Within a two-tier application, there are two computers involved. One is hosting the client and the other one is playing the server role.

Figure **3** is showing the general architecture of a client-server system.

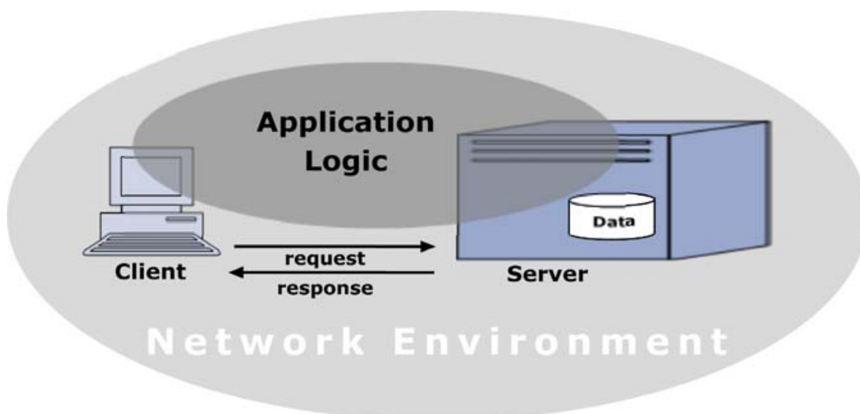


Figure 3: General Architecture of a Client-Server System

Clients - or in other words user agents - are providing the graphical user interface (GUI) for the human-machine interaction. In this case, they are offering possibilities to interact with servers. In order to do so, the client is sending requests to a server and is handling the server's response in form of information representation. Clients are also able to host certain parts of the application itself (see below). In contrast to the client, the server is hosting application logic which is able to handle the client's request. These can include for example running certain threads in a Java environment which are retrieving information from a database. Furthermore, the server is responding the requested information to the client. The communication between the components is based on response/request protocol models. As transport layer the usual types of protocol for certain networks are used. Protocols like TCP/IP offer services for the transportation …

In client-server architectures, the most critical aspect is the balance between the client and the server. This means, a server can provide more or less application logic – the client is behaving correspondingly. Peng and Tsou are calling this act of balance “Client/Server System Partition” (Peng and Tsou, 2003). Therefore, the centroid of the bubble representing the “Application Logic” in
Figure **3** can be relocated horizontally. If the majority of the application logic is hosted by the server it is recognized as “thick”. Consequently, the client can be regarded as “thin”. Similarly, if the server is hosting less application logic, it will be called “thin” – the client in such system would be regarded as “thick”.

Typical examples are web-enabled systems. Here, the client is using a Web browser to represent the data, whereas on the server-side there is a Web-server taking care of the communication. As transport layer, so called Web-protocols like the HyperText Transfer Protocol (HTTP) are used. Other typical examples are the generic database server like Oracle. Here, the data itself it accessible through the database server. The database client can be any application which has the ability to connect to the database server.

The introduction of client-server architectures brought significant improvements. In contrast to mainframe systems in which clients are “dumb”, clients and servers are able to share application logic. Therefore, servers are able to process more requests – even several servers can be in use. This aspect is marking the beginning of decentralized systems (Peng and Tsou, 2003).

However, today we know about the weakness of this kind of client-server systems. First, distributed computing is not fully available since client-side logic is only available for the client itself and not for other users as well. Second, a client-server system is hard to maintain. If software which is hosted client-side has to be updated, every machine has to be refreshed separately. This process can be complicated to organize as well as time and cost expensive. Furthermore, client-server systems are difficult to extend. In some cases, it is even impossible due to the fact that some systems are getting very complex (Nagappan *et al*, 2003).

2.3.1 Component Object Servers

The introduction of Component Object Servers was marking a new period of distributed computing. Its systems which can be regarded as advanced client-server systems are unifying the concepts of Object-Oriented Modeling (OOM) and Distributed Computing Environment (DCE). The concept is proposing that different interfaces are specifying methods which are allowing the use within heterogeneous systems. Therefore, systems based on Components Object Servers are by far more flexible and interoperable than usual client-server architectures. Software components and its objects can be accessed, instantiated and therefore used remotely from other computers. In order to call functions on different machines, communication is based on Remote Procedure Call's (RPC). Roughly, RPC works similar to ordinary function calls in software development (Gisolfi, 2001).

In addition to communicating through the request/response model, many Component Object Server Systems support interaction through "messaging" which is based on passing/queuing. Here, the server and the client do not necessarily have to run at the same time. This is the reason why this type of communication is characterized as "asynchronous". For example, if the server is not accessible, messages are able to be queued until the server can handle them. Therefore, there is a guarantee that the call is delivered (Nagappan *et al*, 2003).

Most famous contributors for the implementation of distributed systems are the open standard Common Object Request Broker Architecture (CORBA), the distributed extension of Microsoft's Common Object Model (DCOM) and Sun's Java RMI (Remote Method Invocation). All of them have been proven in the field of distributed computing. Applications implementing Component Object Servers are generally belonging to the field which is known as "middleware". Systems, using communication through "messaging" are categorized as Message Oriented Middleware (MOM) (Peng and Tsou, 2003).

In contrast to simple client-server architectures, Component Object Servers have brought significant improvements in the field of distributed computing. One important characteristic is such systems are by far more comfortable to maintain than usual client-server systems. Characteristics like scalability, interoperability have made progress, too. Cross-platform applications are possible. However, there are still many things missing which are required in order to offer proper distributed computing. First, Component Object Servers depend on single implementations. Although they have been developed for several platforms respectively different implementations and they claim interoperability. This condition is not reaching higher-level services. Gisolfi is showing the reality and concluding that certain protocols are propriety developed and therefore not interoperable in practice (Gisolfi, 2001). Related to this problem, Nagappan *et al* is pointing out that for example Microsoft's DCOM technology is limited to Microsoft's application environment and therefore platform locked-in (Nagappan *et al*, 2003). Second, many distributed computing implementations are very complex in practice. Very deep knowledge of the specific implementation is compulsory. Therefore existing developing resources can hardly be used. As a conclusion at this stage of the evolution, distributed computing needs standards and agreements in order to become more open and more interoperable (Nagappan *et al*, 2003).

Regarding the topic of distributed virtual worlds, Diehl discusses how to implement CORBA and Java RMI based multi-user worlds. His approach proves that these technologies are able to achieve the tasks of distributed virtual worlds (Diehl, 2001).

2.3.2 XML and Web Services

Recently, the main push towards distributed systems has been contributed by the introduction of the Extensible Markup Language (XML) and very close related to that

“XML Web Services”. In fact, XML-based applications are spreading everywhere. This new family of technologies is used to share data between different applications, platforms or computers. Furthermore, XML Web Services are offering the possibility to share applications throughout a network of computers like the Internet. Therefore, one is able to conclude that XML is offering the possibility to improve the interoperability of information systems. In addition, XML is helping to decentralize those systems. In order to lighten the mystery around XML, the following paragraph is showing some characteristics.

XML is a standard - recommended by the W3C - which is setting rules and guidelines for describing structured data. Perhaps the most important feature of XML is its text-based way of storing data. Furthermore, another characteristic of XML is the strict differentiation between content (elements and attributes), structure (schema) and styling. In fact, the success of XML is showing the need for non-proprietary ways of storing any kind information. Therefore, many XML derivates – which can be standards as well - were developed over the last years. This is the reason why the hype around XML as a keyword should be re-named in the “XML family”. In order to mention a few of these corresponding technologies, there is XSLT (Extensible Stylesheet Language for Transformations) for transforming one XML document into a different structured XML file. Furthermore, there is XSD (XML Schema Definition) to define the structure of a XML document even with following the rules of XML itself. Related to visualize data, there is SVG (Scalable Vector Graphics) and X3D (Extensible 3D) available. In order to define geographic data according to the XML definitions, the OGC introduced the Geographical Markup Language (GML). As stated before, all these are markup languages and are according to the set of rules given by XML. And one can recognize, the XML standard can be used to store/organize/share and visualize data as well as to construct applications. Since XML became a “Web Recommendation” – which means nothing else than “Web Standard” - in 1998, its ideas have been adopted by almost every software vendor.

In order to understand XML and its role within web services

Figure **4** is showing the concept of web services as a simple operational model.

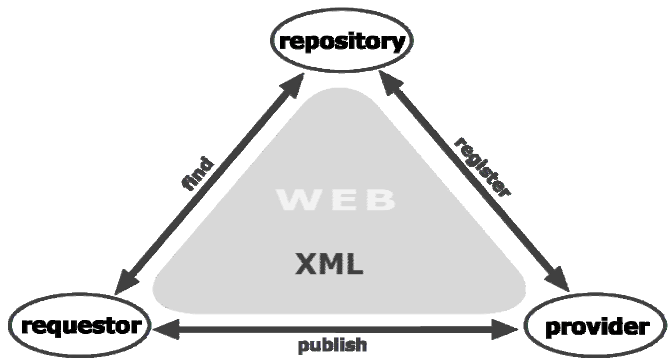


Figure 4: The Operational Model of XML Web Services (based on Nagappan *et al*, 2003; Maruyama *et al*, 2002)

As shown in
Figure **4**, three important components are involved in every Web Service operation. These are namely, the user or service requestor, the repository or registry (service broker) and the service provider. In this model, the client is finding and requesting the service. The repository registers and lists the Web Services with its attributes whereas the provider is processing the Web Service and publishing XML-conform data. Transformed into the client-server model, the user is acting as client and the service provider as server. Because the service broker is located between the client and the server, it is regarded as middleware. In order to implement more complex structures, services can be chained (Open GIS Consortium, 2004b).

In order to share data, XML is the base for any format. As transport layer common Internet protocols like HTTP or FTP can be used. On top of them, the Simple Object Access Protocol (SOAP) is the key standard. As it is a protocol which is standardized and XML-based as well, SOAP is responsible for the communication between the components. SOAP supports both, messaging and request/response communication models. Like protocols in Component Object Server architectures, SOAP also defines ways to perform remote procedures calls. In order to describe and connect to Web Services, the Web Service Definition Language (WSDL) and UDDI (stands for Universal

Description, Discovery and Integration) are used. Whereas WSDL is responsible for describing network services, UDDI defines standard interfaces for dynamically finding other web services. The big advantage of XML Web Services compared to the technologies mentioned before is that it can communicate with any kind of object infrastructure (Coyle, 2002).

Despite the described ways of implementing Web Services, there is the possibility to use ebXML for creating global electronic marketplaces as well. This standardized solution provides a core possibility - by defining the necessary protocols – to implement an electronic marketplace. Therefore, it fits better into B2B processes and it is just mentioned as a sake of completeness (Nagappan *et al*, 2003; ebXML, 2004).

Because all the different protocols and standards sound confusing, unfortunately we still face some problems around XML Web Services. Some protocols/recommendations did not become open standards yet and are not recognized by the main vendors. Even important standardization committees and the software industry sustain a common Web Services model, specific implementations are proprietary developed – one example is the definition of a common security model for XML Web Services. At this stage enterprises even compete and do not cooperate at all (Open GIS Consortium, 2004b).

Nevertheless, in order to realize and host useful Web Services, among others the frameworks of .NET (Microsoft) and Java and its derivates (SUN) are used. Whereas "Microsoft = Microsoft", Java has been adopted by companies like IBM, Oracle and Hewlett Packard. Now, the common thing is that all of them are facing towards XML Web Services (Coyle, 2002).

As a conclusion, one can claim that XML Web Services are providing many important characteristics for implementing distributed systems. In fact, distributed computing is becoming reality. In addition, the GIS community can take advantages out of this development and implement Distributed GIS (see next Chapter).

2.4 3D Spatial Data on the Web: The Development towards Web-Enabled Geo-Informatics

2.4.1 History of GIS: From Monolith Systems to Distributed GIS

30 years ago, mainframe computer systems were offering geo-functionality and geo-data. At this time, only these huge and expensive systems were offering the execution of the complicated tasks of Geoinformatics. Although computer networks have already been involved, these systems are recognized as monolith or centralized systems. All Geo-Information logic and the spatial data where hosted on a mainframe computer. Dumb clients could access the system through a computer network, however too many users were slowing down the processing speed enormously. At this time, such systems were very expensive and therefore only affordable by big enterprises or institutions. This is the reason for a low general accessibility.

In the eighties, concluding to the introduction of Personal Computers (PC), the period of Desktop GIS began. A Desktop GIS consists of a PC which is hosting its own GIS and the data locally. Later, geo-information could be retrieved through computer networks, too. Because of its relatively cheap asset cost, Desktop GIS was spreading out quickly. Henceforth, by far more users were able to take advantage out of GIS. Although every single GIS installation had to be licensed separately. However, the increasing number of individual users was raising new issues. Because GIS were hosted on different platforms and networks interaction became very complex and sometimes was not even possible any more. Especially the communication between applications and sharing data were demanding better concepts of Distributed GIS.

With the introduction of new and more innovative networking technologies – here especially the Internet and its protocols - the situation changed again. As stated before, the progress of IT influenced the field of Geo-Information highly. The terms of Web-GIS, Internet GIS or Distributed GIS are throughout everywhere now. Basically these systems are using the network’s components or objects to store, process and visualize geographic information. Therefore, one can recognize such systems as open and accessible in better ways. The modern characteristics of Distributed GIS are going towards interoperable systems. This means, they should not depend on certain devices, its platforms and further restricted technologies and data formats. However, this development is still in progress (Peng and Tsou, 2003).

2.4.1.1 Distributed GIS

Distributed GIS is describing a selection of GIS nodes or objects using computer networks as the primary medium to access, manage, analyze and visualize geographic information. The objects together are achieving a GIS comparable to a centralized desktop or mainframe systems. However, the physical aspect is completely missing since the components are used within a networking environment. In addition, characteristics of Distributed GIS are accessibility and extendibility – experts are talking about interoperable systems, too. Main intentions are the integration of spatial information in different systems. This is including the dissemination of GeoServices and the strong need of sharing geo-information so everyone can benefit. In contrast to other “networking GIS” like webmapping in usual client-server architectures, distributed GIS is taking all the advantages of state-of-the-art distributed computing. Therefore, the most recent technology of XML Web Services fits nearly perfectly. Figure 5 is showing the scenario of a Distributed GIS realized by using the OGC Web Service model.

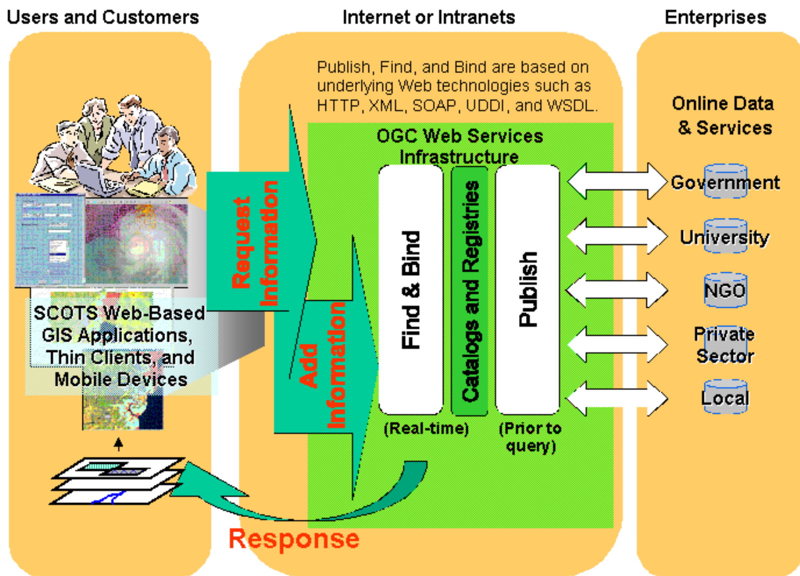


Figure 5: Distributed GIS scenario using the OGC Web Service architecture (Open GIS Consortium, 2004b)

The vision shown in Figure 5 is proposing a system which can be accessed by users from any device and many different applications – independent of its location. Furthermore, the server should provide certain GIS components which are fitted into a Web Service infrastructure. Corresponding objects are offering methods to access different data sources and to process geo-information, e.g. to perform spatial analysis. Therefore, it does not matter how complicated the processes are. They can be either a simple coordinate transformation or complicated spatial analysis. The geo-information community is calling these services as “GeoServices” (ESRI, 2001) or “GIServices” (Peng and Tsou, 2003) in favor. In order to provide a useful application which is capable to publish, find and bind, many GeoServices even on different machines have to be integrated. In theory, there are no restrictions regarding the implementation of Distributed GIS. However, the reality is telling us that performance, stability, security and privacy are most critical and still barriers on the track to Distributed GIS (Open GIS Consortium, 2004b).

Because many people are concerned about these critical aspects, general questions like "How can anybody benefit from Distributed GIS?" are occurring – here, anybody is including the GI-community as well as the public. In order to find the right answers – besides inheriting the advantages of distributed computing in general (see Chapter 0) -, problems appearing in current GIS solutions have to be realized. At the moment, one is able to differentiate between several causes of non-interoperability which are isolating spatial data. These can be summarized into technical and semantic non-interoperability. Whereas the first one is facing propriety GIS solutions including internal data formats, the later one is dealing with problems around internal data structures and schemas. All of them can be solved by developing web-orientated GIS. Beside general web standards, special "geo-related" rules and standards have to be settled to achieve interoperable systems. Most recently, standardization organizations like the OGC or the ISO/TC 211 (ISO/TC 211, 2004) are working hard to define the necessary spatial standards (see next section). If the community – here especially the main GIS vendors – are taking care of these regulations, the chances are very high to establish real web-enabled GIS. The general tendency is more and more GIS software is implementing standards and working towards interoperable systems. Hence, the OGC is predicting the so-called "Spatial Web" (Open GIS Consortium, 2004b). As one important result the value of geo-information and its technologies will be increasing. Everybody in the field of geo-information can take profits out of the development towards web-enabled interoperable systems (Open GIS Consortium, 2004b).

Beside these theories, certain components of Distributed GIS have already been established. Whereas – obviously - there is no recent overview about internal enterprise solutions, among the most popular examples are Microsoft's MapPoint Web Service (Microsoft Corp. 2004) or ESRI's Geography Network portal (ESRI, 2001). However, web services implementing the third dimension –here especially urban environments – have not been fully realized so far (the discussion about realized approaches, see Chapter 0).

Moreover, the following thoughts are providing some expressions which communities are able to benefit from Distributed GIS. Most recently, especially e-government scenarios that are proposing Spatial Data Infrastructure (SDI) can take advantages of interoperable systems. Distributed GIS is regarded as a reasonable way to achieve SDI. In many countries and even across continents, there are ongoing efforts to disseminate spatial data in order to use them. Since the Web became the dominant platform even a global SDI is not an illusion anymore. Common scenarios of a Spatial Data Infrastructure are including connected GI nodes which are provided by government agencies, universities, enterprises, organizations and others. The different “data provider” updating and maintain the certain information. Such a model is proposing a decentralized data management. Furthermore, the distributed objects spread the spatial data and offer certain geographic processes like spatial analysis as well. Any constellation could be possible. A common Spatial Data Infrastructure obviously has many advantages. First, costs can be cut for example through the reuse of data. Therefore it is not necessary any more to spend that much money and time on data acquisition by several parties at the same time. If more and more people are using the data, it will become cheaper and cheaper, too (Lake, 2004). Furthermore, the dissemination of spatial data is the base for many useful applications. Not only the GI-community is taking profits out of SDI, moreover everyone would benefit from better accessible data. Here, location services and street routing for car navigation are typical example which the public has already accepted and using of spatial data. Third, the advantage of higher GIS customization in terms of usable GI-modules is obvious as well. Since distributed computing is based on modularity GIS benefits out of this characteristic. Whereas today, core monolith software kits have to be purchased even if only one small component is needed, Distributed GIS are offering the possibility to purchase only the needed components (Peng and Tsou, 2003).

2.4.1.2 Open Geo-Standards and their Value for “The Spatial Web”

Since components in distributed systems are likely to be heterogenic, open standards to communicate properly are most important. In order to call a standard open important characteristics have to be fulfilled. Reichardt (2003) is introducing them in detail. Among others, the most important ones are:

Standards should be open and accessible freely by everyone. Furthermore, standards should be created by non-propriety and not-profit organizations. In most cases, these are consortia including main vendors, government agencies and research institutions as their members (Reichardt, 2003).

The reasons why open standards are necessary should be obvious to everyone, too. Beside the requirements for general agreements in distributed systems there are other side effects. In the past, GIS vendors have competed to establish their products and its linked data formats. As a result of establishing standards, vendors can concentrate more on security and other application aspects like analysis (Lake, 2004).

In the past, client-server systems – especially vendor based webmapping solutions – were facing a lack of interoperability. Kolodziej talks about “technology islands” and describes a corresponding scenario (Kolodziej, 2003). Frankly, these islands were not able to communicate and therefore to share information between them. However, he points out that the situation has already improved (Kolodziej, 2003).

As a conclusion, for distributed systems standards or common agreements are a must. As stated before, it is known that different platforms and applications are “troublemakers” in terms of sharing data and resources efficiently. Beside the general computing standards, geo-information needs its specific rules for defining characteristics like data semantic, interfaces and processing services. Furthermore, architectures for the concrete implantations are important. Therefore, mainly two standardization organizations have to be recognized. First, the recommendations and specifications developed by the OpenGIS Consortium (OGC) are considered as common standards. Furthermore, the ISO/TC211 and its standards are widespread as well. These two organizations are the key players in the field of Geo-Information. Both are cooperating in order not to compete with each other (Oestensen, 2001). Peng and Tsou (2003) are giving a detailed overview about the overlapping and different parts of the specifications. In relation to this research (see next Chapter), Oosterom *et al* are pointing out that both standards are well harmonized in terms of defining the same geometric and topological primitives (Oosterom *et al*, 2002).

The reason for different developments can be found in the background of these two committees. Whereas the ISO/TC211 has its members in the public and government sector, the OGC is a consortium by mainly companies and institutions. Because of this different background, the history of the standard development was different. The OGC is mainly focusing on interoperable systems which are combining existing GIS solutions. The ISO/TC211 has its origin in constructing a Spatial Data Infrastructure for nations and certain regions (Peng and Tsou, 2003). In the case of this research, mainly the specifications of the OGC will be recognized. The reason for this is, nearly every important company and many research institutions are members of the OGC – therefore it is recognized as the de-jure standardization organization in the field of Geoinformatics. Beside, as stated before OGC and ISO/TC211 are cooperating more and more and the standards are even merging.

The OGC is categorizing its catalog of specifications into two groups. First, there are the OpenGIS Abstract Specifications. Its purpose is to define conceptual models in order to specify implementation interfaces. The different abstract specifications are divided into 16 topics. Among others, the most important topic of the OpenGIS Abstract Specifications for this research is “Topic 1”, the “Feature Geometry”. The corresponding “Simple Feature Specification” is defining the schema of geometric features and their topological relationships.

Second, the OpenGIS Implementation specifications are focusing on software specifications which are adopting the OpenGIS Abstract Specifications. Therefore, software vendors are able to propose their specifications. Table 1 gives an overview about the research related OpenGIS Implementation Specifications which already have been accepted by the technical committee of the OGC.

Table 1: Research Related OpenGIS Implementation Specifications

Volume Name	Explanation
Web Map Service (WMS) Implementation Specification	Part1: defines the interface to request maps from a server Part2: discussion paper. It is facing towards XML-enabled map service implementations
Web Feature Service (WFS) Implementation Specification	Web interface definition for inserting, deleting and manipulating data
Web Coverage Service (WCS) Implementation Specification	service for accessing raster images
Web Terrain Service (WTS) Implementation Specification	extend the WMS with 3D terrain functionality like elevation models, texture images and viewpoint definitions – raster-based
Simple Feature Specification for SQL	defines a SQL schema to interact with simple features stored in a database
Geographic Markup Language (GML) Implementation Specification – Version 3	specified XML-based exchange format for geographical data

Whereas the first four ones defines methods and properties to request and serve data between a client and a server, the fives one standardizes the interaction with a spatial database. In addition, GML is a standard for sharing and exchanging geographical datasets. Regarding the system design and construction, the OGC is proposing its own vision which is based on a client-server architecture including XML Web Services (see Figure 5). At the moment, the implementation specifications of Web Map Service, Web

Feature Service, Web Coverage Service and most recently the Web Terrain Service are not only separately existing specifications. Furthermore, the can be used complementary (Kolodziej, 2003). Kolbe and Altmaier are showing one possible system architecture for using these services. Furthermore, they discuss the raising issues by implementing 3D worlds using existing OGC Implementation Specifications. As a result, they propose the W3DS portrayal service for 3D spatial data. The reason is, because so far there is no standard to visualize interactive 3D models. The WTS is restricted to static images only. Furthermore, there is no standardized way to implement necessary 3D features like backgrounds or characteristics of the atmosphere (Kolbe and Altmaier, 2003).

In order to realize the vision of the “Spatial Web”, the OGC has set up the OGC Web Service Phase 2. The intention is, to build up a core framework of implementations in order to achieve interoperable GIS (Open GIS Consortium, 2004a).

Beside the specifications and initiatives, the OGC is publishing other documents like discussion- and white papers as well as reports. The main purpose of them is to introduce and explain new thoughts and technologies related to geo-information. In order to cover all necessary standards for this research, the certain specification will be mentioned within its specific chapter.

2.5 3D Spatial Data Management

The data management considers mainly where to store and how to organize geo-information. In contrast to the 3D data modeling (see below) it is not necessarily related to web-oriented systems. The following paragraphs give a brief overview about general database concepts theories and its use in the field of Geoinformation. Furthermore, tasks of spatial data management with the interests in the third dimension will be discussed.

2.5.1 Database Concepts

Any complex GIS project consists of a Database Management System (DBMS). The reasons why DBMS are more appropriate have been discussed for decades already. Connolly and Begg (2002) have provided a brief overview why DBMS have many advantages compared to the data storage within files (Connolly and Begg, 2002). Databases offer a more efficient way to store data. Characteristics include the control of data redundancy which is reducing the risk of inconsistency as well and the possibility of sharing information. The later aspect is very important because DBMS are offering multi-user accessibility. In general, the data accessibility is improved through querying e.g. through the Structured Querying Language (SQL). Therefore, data manipulation processes like updating and deleting are by far easier to achieve. Important side effects are the positive querying performance, data security/backup possibilities and the integratability into different system environments. Although DBMS have some disadvantages like complexity and high costs, the advantages the field of Geoinformation can take are preponderating. As stated before, complex GIS projects request the storage within databases and a proper management of them.

Although the diagrams are good ways to keep the track of concepts, intricate models can become very complex and obscure. Furthermore, Rigaux *et al* points out that the relational model is not suitable to host geographic data properly because of its inherent spatial component. Whereas usual business applications are containing large but simple datasets, geographic information has the characteristic to be large and very complex (Rigaux *et al*, 2002).

In contrast to the relational model, the object-oriented (OO) approach to model databases is using a completely different concept. As the name implies – the corresponding expression for its management systems is OODBMS-, this model is adopting object-oriented concepts. Therefore the conceptual model is defining objects and its properties as well as the corresponding behaviors called methods. Objects are able to inherit from other objects – an object model has the shape of a tree. The logical model is creating instances and is populating them with information. The motivation for the development of object-oriented database models is obvious. Since state-of-the-art software development adopted OOP, the need for keeping data in similar ways on database level increased. Whereas relational database models have to be transformed into the object models of applications in order to achieve GIS, the object-oriented conceptual database schema can be mapped directly to the object-oriented application. Since this transformation is chore, object-oriented databases have a big advantage compared to relational ones (Shekar and Chawla, 2003). Furthermore, Franklin points out, that object-oriented models are by far more elegant to design than relational ones (Franklin, 2001). Shekar and Chawla add that especially spatial entities like lakes, road networks and cities are perfectly fitting in an object-based approach (Shekar and Chawla, 2003). Gruber is going one step further and claims that state-of-the-art urban data management deals with complex data and requirements and therefore has to replace the relational model on database level (Gruber, 1999). One example for the implementation of an object-oriented data model has been developed by Abdul-Rahman (2000). Briefly, he converted the Pilouk's Tetrahedral Network (TEN) into an OODBMS (Abdul-Rahman, 2000).

However, in association with the object-oriented approach there are several disadvantages as well. These are mainly the missing overall accepted querying standard and the lack of experience. Other problems like the lack of implementations are associated to them. On the market of OODBMS, there is less competition compared to the one of RDBMS. In order to improve the quality of the implementations, fair and open competition is one fundamental aspect (Connolly and Begg, 2002). Another disadvantage is that many applications have already been realized in a RDBMS and their migration to OODBMS is among the biggest challenges (Franklin, 2001).

Applications of Geoinformation are using both concepts. Whereas more simple projects are fine with adopting the relative model, more complex spatial data is able to benefit from the object-oriented as well. Thus, the modular approach – named object-relative database - has become popular in the field of Geoinformation over the last years. Shekar and Chawla are proposing an architecture including an object-related database management system (OR-DBMS) for the use within GIS. They circumscribe it by the term of "Spatial Database", which basically is a database extended by spatial data types and associated functionality (see below). This is most appropriately done by using an object-relative database backend (Shekar and Chawla, 2003). In this context Rigaux *et al* talks about an integrated approach (Rigaux *et al*, 2002); Oosterom *et al* propose the integrated architecture as well; in conjunction with a corresponding DBMS they are using the term “Geo-DBMS” (Oosterom *et al*, 2002). Object-relational databases are offering the possibility to define individual spatial data types as well as implementing pre-defined data types through objects. These objects will be stored in the cells of a certain table. The tables are located in a relational model. The objects are regarded as the extension of relational databases. The advantage of using objects as data types compared to store spatial data for instance in a pure relational model is obvious. The spatial information becomes handier and complicated transformation can avoided in favor. Furthermore, certain standard object behaviors are represented by an object’s methods. Thus, typical spatial operations like an area calculation can be performed on the database level rather than developing certain functionality for each system. In this context, Connolly and Begg are circumscribing this characteristic with the terms of “reuse” and “sharing”. Among usual database requirements, spatial extensions should implement spatial indexes (Shekar and Chawla, 2003) as well as topology in order to offer reasonable ways to query data. As a conclusion one can claim that advanced data management applications - like GIS - are able to benefit from OR-DBMS (Connolly and Begg, 2001).

However, there are a couple of problems which have to be considered. First, OR-DBMS can become very complex and obscure. Beside, most problematic with the use of spatial OR-DBMS is the missing common standards. At the moment, still some inconsistency between the object-relational extensions of different database vendors remains. Here, the missing implementation of a standardized querying language has to be mentioned. Although SQL3 - which standardizes for instance the creation of individual abstract data types – has been published long time ago (in 1999), many of the current implementations are using proprietary techniques. However, SQL itself is not covering any “spatial” features since it is developed for general querying purpose. Therefore, the OGC published a recommendation - Simple Feature Specification for SQL - for extending SQL (see before). It is defining certain spatial data types and functions for spatial data querying (Shekar and Chawla, 2003). Oosterom *et al* points out that only simple features are covered. The recommendation – or in OGC language the implementation specification - for abstract features like sophisticated topology is still missing (Oosterom *et al*, 2002).

Comprising different models to store data, the four-quadrant view proposed by Stonebracker, 1996 is very useful (see

Figure 6). Whereas the x-axis is reflecting the degree of data complexity, the y-axis shows the search as well as the multi-user support possibilities.

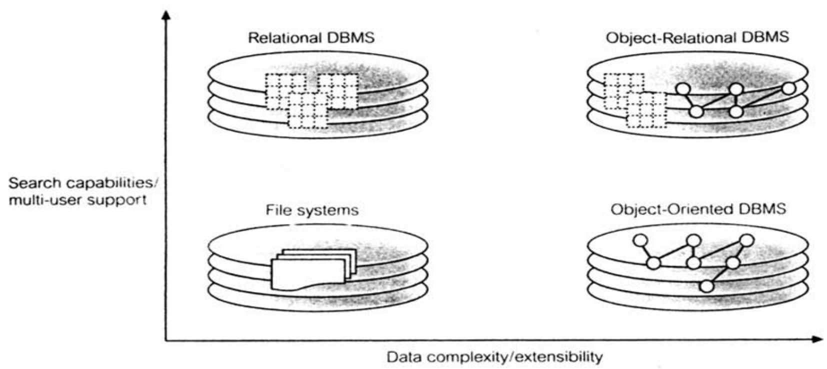


Figure 6: The Stonebreaker View (Connelly and Begg, 2001)

Referring to

Figure 6, GIS applications are fitting best in the object-relational approach. In fact, spatial data is in most cases intricate and has to be updated frequently. Furthermore, multi-user support is a requirement for GIS as well.

The following paragraph deals with the integration of 3D spatial data in these database models.

2.5.2 Spatial Databases and 3D Geo-information

Spatial databases which are gathering the third dimension are considered as 3D Spatial Databases. Therefore, the Spatial-DBMS (SDBMS) should offer a core implementation of storing 3D geo-information. The general requirements for spatial databases can be defined as follows (based on Güting, 1994):

- offering spatial data types and its implementation
- providing a corresponding query language
- creating spatial indexes
- processing of spatial analysis

Oosterom *et al* extend these requirements by the topological aspect (Oosterom *et al*, 2002). The requirements are at least valid for multi-dimensional data as well.

In order to look closer on 3D spatial data, Geo-information has to be divided into the geo-data itself and their associated attributes. Since storing attributes is not that critical, a closer look at the geo-information itself is necessary. As mentioned before, the aspects of 3D geometry and 3D topology have to be considered while dealing with 3D spatial data. This is a must because spatial functionality strongly requires them as a base for calculations (see 0). Compared to the second dimension, 3D data is much more complex. Whereas 3D geometry is easy to implement, 3D topology is most critical. In order to implement topology spatial operations are necessary, too. These important topics will be discussed in detail later (see below).

The second requirement is the provision of an associated querying language. Since SQL is a general querying standard, there is no feature for spatial querying. Thus, the geometrical and spatial part has to be integrated. In General, a spatial query language has to provide fundamental spatial operations and reasonable ways of representing the result (Güting, 1994). Egenhofer adds that the combination of both, spatial and non-spatial types of queries is a basic guideline for a spatial query language implementation

(Egenhofer, 1994). Beside the query, especially the representation of the result has to be considered (Güting, 1994). Most important, standards for accessing, querying the database and displaying the results are important. Thus, in 1999, the Open GIS Consortium has set up the "Simple Feature Specification for SQL" which is proposing spatial data types and operations (see before). The later one can be categorized into basic operations for all geometric data types, operations for simple topological queries and general operations for spatial analysis. However, the standards lacks of sophisticated integration of topology and metric spatial relationships like directional queries (Shekar and Chawla, 2003). Furthermore, query results should be represented in standardized ways as well – e.g. GML as a XML derivate can be used. In the practice, spatial querying languages are implemented through the DBMS. Therefore, examples are provided through Oracle Spatial or PostgreSQL/PostGIS.

In order to support spatial selections the use of spatial indexes is required. A spatial index - as indexes in general - is taking profit of approximations. The index structures data into simpler geometric objects than the object itself by creating spatial keys. During the query process, index values are used to browse the data faster. There is a large variety of spatial indexes available (Shekar and Chawla, 2003). One good example – especially for real 3D objects - is the use of 3D bounding boxes, called "R-Tree". In the third dimension a bounding box is representing the smallest axis parallel cuboid enclosing the object. The R-Tree is constructing a hierarchical tree of these boxes. The science of databases mentions different variants of R-Trees. They are mainly differing in their grouping strategies (Kofler, 1998). In conjunction with 3D visualization, R-trees are common in order to implement hierarchical levels of details (LOD). Therefore, the index structure is used to represent different levels of detail (Coors, 2003). Among others, Kofler (1998), Zlatanova (2000) and Coors (2003) have proven that R-trees are a proper method to implement hierarchical LOD. Whereas Kofler (1998) implemented the R-tree in an OODBMS, Coors (2003) and Zlatanova (2000) chose the relational and object-relational model to do so. Another indexing method is called Quadtree or in the case of the third dimension Octree. This approach is based on tiling the space into a grid of cubes. In contrast to the basic R-Tree, 3D objects of an Octree are not overlapping. Whereas R-

Trees can be used in order to implement LOD, Octrees are mainly utilized to speed up spatial queries. Besides, the object-relational database “PostgreSQL” and its spatial extension PostGIS are implementing its own indexing method called GIST (Generalized Search Trees). However, it has been not proven for the third dimension. Because of the PostgreSQL page size, objects larger than 8K are not supported in R-trees (Ramsey, 2003).

The third requirement is the integration of spatial analysis and operations. Therefore, the database level should provide basic GIS operations like spatial selection, spatial join, spatial function application like intersect and other spatial set operations. Whereas the first two are not that critical and can be achieved by usual queries, the later tasks have to be considered more complicated. They do not fit into a “SELECT … FROM … WHERE” routine because they are on a more abstract level of spatial operations. As stated before the “Simple Feature Specification for SQL” is standardizing the most common operations. An overview can be found by Shekar and Chawla (Shekar and Chawla S., 2003). The task of spatial analysis - linked to the associated querying language as well - is already reaching application level. Many GIS application are implementing these methods already by itself (Güting, 1994). However, providing common tasks already on database level has many advantages. Overall, an intelligent implementation can improve the performance of certain processes. The reason for this is because the user access on the database will be reduced and therefore the critical bottleneck of the database interface is bypassed (Jansen, 2003). Furthermore, the term around Geo-information is becoming more accessible without purchasing specific and expensive GIS. As a conclusion, a more simple system integration of Geo-information is one major advantage.

Today, spatial database implementations are wide spread. Nearly every main database vendor is providing spatial extensions in order to penetrate into the GIS market. However, Stoter and Zlatanova point out, that current implementations are not yet fully compatible with the third dimension (Stoter and Zlatanova, 2003a). Whereas the second dimension has been integrated in proper ways – 2D spatial data types, 2D geometry, partly 2D topology as well as 2D spatial indexes-, 3D spatial data is not fully integrated yet. Unfortunately not many DBMS are offering support for 3D indexing – however, Zlatanova (2000) shows that R-trees can be implemented into non-spatial databases as well. Furthermore, the integration of 3D topology is still missing (Stoter and Zlatanova, 2003b). However, Oracle recently announced the integration of topology and R-trees up to 4D in its database Spatial Extension of Oracle 10g (Lopez, 2003). In addition, Ravada is giving a detailed introduction about Oracle's topology implementation (Ravada, 2003). On the research side, Arens (2003) is extending spatial databases with an individual spatial (volumetric) data types, namely the 3D polyhedron primitive (Stoter and Oosterom, 2002; Arens *et al*, 2003).

In conjunction with the management of 3D data, the possibilities of manipulating data including the creation, the edition and the query of stored information is significant as well. Here, proper access to the DBMS is most important. The database interface can be an important bottleneck in systems – especially once one is dealing with the third dimension. Furthermore, applications have to be provided in order to create the spatial data types and populate the database. Especially the complexness and abstractness of 3D geo-data are requesting reasonable graphical user interfaces therefore. Although, the newest product lines of CAD respectively GIS software are supporting these tasks quite well, there is no implementation which is using a real web environment (Stoter and Zlatanova, 2003b). For instance, Nebiker presents the DILAS approach which is offering 3D visualization on the Web (see below). However, the data edition is only possible through an Intranet solution (Nebiker, 2003).

Finally, in order to transform real-world 3D objects into database system different concepts for 3D data modeling have to be used. The following paragraph is giving a deep introduction about recent literature and research around the concepts of 3D data modeling.

2.6 Concepts for 3D Data Modeling

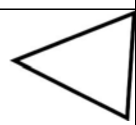
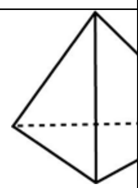
3D data modeling deals mainly with certain ways to transform real-world 3D objects into spatial databases. Therefore, specific concepts have been introduced and several implementations have proven to be sufficient for modeling real-world 3D objects within spatial databases. Without regarding the dimension, there are several possibilities to store the geometry of spatial objects.

First, the Spaghetti Model is able to hold spatial objects. Here, the geometry of a vector-based object is stored in collection which can contain any kind of feature like points or lines. Furthermore, objects are stored independently of others. Therefore, data redundancy is not implied. For instance a line used by several objects is represented twice or more. The only advantage is its simplicity. Geo-information modeled as "Spaghettis" can be extended easily and the reconstruction in order to visualize is not critical as well. However, very limited GI-functionality can be processed on top or the data has to be re-modeled to other models like the topological one (Rigaux *et al*, 2002). Concluding, data stored in the Spaghetti Model can be regarded as "dumb" and therefore less useful for any spatial analysis.

Second, the topological model is most important. As stated before, topology is dealing with the relationships between spatial objects. Due to the fact that geographical data can be raster- or vector based we have to distinguish between two completely different kinds of topological concepts. First, and less important for this research, there are the topological relations between raster cells. Molenaar stated that these are necessary for window and filter operations (Molenaar, 1998). Second, 2D topology of vector data is based on the geometrical objects of point, line and polygon objects. Extending a topological model to the third dimension, a body object has to be introduced. Each element is represented by its predecessor. For instance, a line is represented by two points - a body by at least three polygons. Overall, GIS and especially its analysis part require topological data models in order to perform efficient. Previous research showed that different ways for topological modeling are possible (Rigaux *et al*, 2002). The following paragraphs of this chapter are giving a detailed overview about realized topological models.

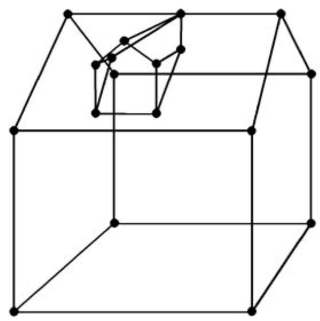
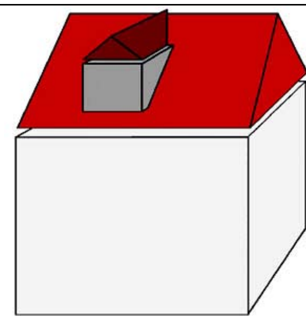
In order to get used to common expressions around 3D modeling of geographic information, Table 2 gives a brief overview.

Table 2: Expressions according to their Status Modeling Level

Graphical Representation	**Geometric / Elementary Objects**	**Primitives (Constructive Objects)**	**Topological Primitives**	**Definition Level**
	point	node	node	0-simplex
	line	arc/edge /curve	edge	1-simplex
	surface	face	face	2-simplex
	body	edge/face	solid	3-simplex

All objects shown in Table 2 are considered as "simple objects" or short as "simplex". Molenaar defined certain rules for simplexes (Molenaar, 1998). Complex features - in a more urban context – for example houses or buildings can be modeled in different ways. In order to avoid application-dependent ways to store 3D objects, several concepts have been discussed before. Pfund is categorizing them as "Solid Models" (Pfund, 2001). Table 3 represents the most important ways for modeling urban objects like buildings in 3D GIS.

Table 3: Modeling Concepts in 3D GIS (based on Pfund, 2001)

Boundary Representation (B-REP)	**Constructive Solid Geometry (CSG)**
	
The object is described through its boundary elements like edges and faces. These are represented by vector data.	One 3D object is modeled with other objects which are on a hierarchical lower level. Certain operators are connecting the objects to a 3D object.

The B-REP approach is using common objects like nodes, edges and surfaces in order to model buildings. In order to include topological relations, the body object has to be added. The B-REP approach is becoming very close to proper implementation of a topological model. Only some more constraints have to be added in order to fulfill the requirements of a 3D GIS. In contrast to the B-REP, the CSG is using space primitives in order to construct buildings. These are geometrical elements like cuboids, cones, spheres etc. The CSG can not host topology – it has to be computed every time (Pfund, 2001).

In order to model 3D objects, the OGC is proposing a differentiation between geometrical and topological models. The next paragraphs about 3D Geometry and 3D Topology are giving an introduction what has already been done in past research.

2.6.1 3D Geometry

The geometry of an object is mostly composed by its coordinated. Here - dealing with the third dimension - the three coordinate values of x, y and z are necessary to describe an object. In reality, most of the available DBMS are supporting the common geometric objects like point, line and polygon. Because, in most cases these are only 2D objects predefined, 3D objects have to be represented by multiple 2D objects with 3D coordinates. In practice, one should be able to differentiate two possibilities for the implementation. First, 3D objects are represented by simple geometry types, i.e. a set of polygons represented by 3D coordinates. Second, one 3D object can be represented by a collection geometry type, for example 3D collection and 3D multi-polygon. Spatial operations implemented on top of a geometrical model can be area, volume or length computations. However, most of them are performing slow compared to the processing on top of topology (Zlatanova *et al*, 2003).

2.6.2 3D Topology

As stated before, topology is a must in this research because real 3D or multi-dimensional functionality on top of the data has to be integrated - geometry is only able to host very limited analysis functionality and corresponding calculations are computationally expensive. In general, topology is dealing with combinatorial structures or relationships between objects. Therefore, topology is used to convert constructional geometric algorithms into combinatorial processes (Open GIS Consortium, 2003). Most common spatial operations which require topology on 2D data are for instance network analysis, intersecting or union. In order to achieve these on 3D geo-information, solving the critical aspect of an accurate topology management is a requirement for spatial operations. Whereas geometry is represented simply by an objects x-, y- and z-coordinates, topology has to be defined in different ways. Therefore, each topological model has it own conventions. Moreover, these directives can be very complicate, too. As stated before, base for most topological models is the Boundary Representation (B-REP). Despite of the fact that the literature mentions other frameworks to define spatial relationships as well, the common conclusion is that the topologic one is the most appropriate for geo-related tasks. The reason for this is its foundation on the neighborhood of the objects regardless of the distance between them (Zlatanova *et al*, 2004).

In order to realize a topological model, general frameworks of topology have to be utilized. In the literature, there are mainly two mathematical frameworks for defining spatial relationships available. First, there is the 9-intersection model introduced by Egenhofer and Herring (1990). This popular model can be regarded as the standard considering spatial relationships since it has been adopted by the OGC (OpenGIS Consortium, 2001). Here, spatial relationships are determined by distinguishing empty and non-empty intersections of the topological primitives. Overall, eight relationships between 3D and 3D objects are possible. These are disjoint, meet, contains, covers, inside, covered by, equal an overlap. As a conclusion, this framework can be regarded as systematic and simple. However, not all the relations make sense in reality and can be realized. Furthermore, the basic spatial object has to be very simple, e.g. without holes and intersecting parts (Billen *et al*, 2002). Second, Billen *et al* introduced the Dimensional Model. Originally introduced as a model for dealing with convex spatial objects, it has been extended to topological n-manifolds. It uses a complete different approach of defining spatial relationships. Whereas the 9 intersection model is based on the intersection of topological primitives, the Dimensional Model is looking at the in intersections of the dimensional elements. Three groups of relations can be provided – the total-, partial- and non-existent relation. The Dimensional Model offers by far more possibilities of spatial relationships and the framework is more flexible compared to the 9-intersection model. However, similar to the 9-intersection model only a few of the theoretical relationships can be realized in practice (Billen *et al*, 2002).

As stated before, topology is mentioned within several OGC abstract specifications. For example the OGC has been adopted the 9-intersection model in order to define a topological framework. However, there is no implementation specification for a DBMS environment like it exists for geometry (Oosterom *et al*, 2002). Previous research is providing us the following topological models: 3D FDS (Molenaar, 1990), cell tuple model (Pigot, 1992), TEN (Pilouk, 1996), SSM (Zlatanova, 2000), SOMAS (Pfund, 2001), UDM (Coors, 2003), OO3D (Shi *et al*, 2003). Table 4 gives an overview about different implemented topological models and their characteristics.

Table 4: Characteristics of Realized 3D Topological Models (based on Oosterom *et al*, 2003)

Name	Primitives	Topological Tables	Explicit Relationships
3D FDS	node, arc, edge, face	arc, edge, face	node-on-face node-in-volume arc-part of-line arc-on-face
Cell-Tuple	0-cel, 1-cell, 2-cell, 3-cell	cells	no
TEN	node, arc, triangle, tetrahedron	arc, triangle, tetrahedron	triangle-part of-surface arc-part of-line
SSM	node, face	face, line, surface, body	node-in-volume face-in-volume
SOMAS	vertex, edge, face, body	point, line, polygon, body	unknown
UDM	node, face	point, line, surface, volume	no
OO3D	node, segment, triangle	point, line, surface, volume	unknown

Table 4 gives an overview about the most common 3D topological models in chronological order due to their publication - the latest one can be found on the bottom of the table. Be aware of that the column “No. of Tables” can differ between certain implementations of one model. The following paragraphs are describing each model briefly (according to their description in certain publications):

The 3D Formal Data Structure (3D FDS) by Molenaar (1990) is based on a single-value map. Here, the space is divided into non-overlapping objects. In order to achieve this, 12 conventions are setting the frame in order to model data correctly. For instance, arcs and faces cannot intersect. The model consists of three fundamental levels, namely feature, elementary objects and primitives. Therefore, it is possible to integrate attributes, e.g. textures of face. However, one face has to be textured with exactly one image – the common task to drape one image across many faces can not be realized. Among others, Zlatanova is facing some further issues, e.g. the missing explicit relation face-part-of-body (Zlatanova, 2000). Coors (2003) added that the storage of edges is not necessary since most topological queries for city models are based on adjacent edges (Coors, 2003). Overall, the model has proved to be suitable for urban applications (Zlatanova, 2000).

The Cell-Tuple model, introduced as an extended version by Pigot (1992), defines cells and cell complexes on top of the fundamental properties of a manifold. The k-complex is the union of all the k-dimensional and lower cells, whereas k is the dimension of the cell. For complex real objects, this model can lead to huge representations. Therefore, it can reach unmanageable amounts of data for complex city models (Zlatanova, 2000).

The TEN model by Pilouk (1996) is using triangles as its base in 2D. The corresponding body in the third dimension is the tetrahedron. Therefore, every object has to be divided into a set of tetrahedra. Shi *et al* realized that this process is very critical and especially architectural objects, like buildings are difficult to model. By the way, a TEN for buildings is causing big amounts of data. The TEN is especially useful for computation and querying which are applied in geologic applications (Shi *et al*, 2003; Pilouk M., 1996).

Zlatanova (2000) introduced the Simplified Spatial Model (SSM). The basic objects are point, line, surface and body. As primitives, the model is restricted to node and face only. The relations arc-in-face and face-in-body are explicitly defined in order to fulfill the requirements of complex urban objects. Furthermore, the model was specifically designed for web-oriented applications with a strong tendency to visualization queries. (Zlatanova, 2000).

The prototype of the 3D vector-GIS called Solid Object Management System (SOMAS) has been presented by Pfund (2001). The model is using the primitives of vertex, edge, face and body in order to store the geometry of a 3D object. Within SOMAS, the data model is transformed into a relational database. Therefore, data redundancy is kept very low only a small number of topological relations are saved explicitly (Pfund, 2001).

Coors (2003) proposed the Urban Data Model (UDM) which is quite similar to the SSM. However, it has been developed separately. In this model, edges are not explicitly stored. Therefore, one-dimensional constructive objects like edges are not supported. However, they can be modeled by defining two successive points. Thus, only polyhedra can be represented by the UDM. Furthermore, "additional representations" (Coors, 2003) or attributes are stored different from the geometry. The advantage of UDM is that is an effective way to store data. Concluding, the implementation – which is web-based - of this query-orientated model has proven that the Internet is a possible medium to access 3D city models (Coors, 2003).

Shi *et al* (2003) proposed an object-oriented data model for complex objects in 3D GIS called OO3D. It is using object-oriented data modeling concepts for storing the information. Furthermore, it has been implemented in a 3D GIS called SpaceInfo. However, on the logical level, the OO3D has been implemented as a relative database model (Shi *et al*, 2003).

Overall, Shi *et al* (2003) and Zlatanova *et al* (2004) provide brief comparisons of the mentioned topological models even with additional performance tests.

2.7 3D Geo-Visualization on the Web

The preceding chapters have discussed how to manage and organize data respectively how to model 3D objects for urban environment. These are the base for a reasonable visualization of 3D city models.

First, the overall process of visualizing data has to be regarded. Therefore, the OGC introduced the visualization pipeline (
Figure **7**) – based on Haber and McNabb (1990) (Doyle and Cuthbert, 1998).

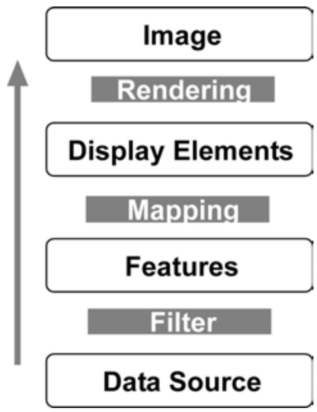


Figure 7: The Visualization Pipeline (based on Haber and McNabb; 1990; Doyle and Cuthbert, 1998)

As shown in
Figure **7**, the visualization process is divided into several stages. At the bottom, there is the "Data Source"-level which is including the data itself and managing functionality. The second stage is the "Features"-level. Here, the data is already pre-assembled by running through a filter like a data query. At third, there is the "Display Elements"-level. At this stage, the data has already been reconstructed to the requirements of the visualization format. For instance, styles have been added to the objects – this process is called "Mapping". Last, on the "Image"-level the data is rendered in conjunction to the characteristics of the device display.

Second, in order to regard visualization in a web-enabled environment, this process has to be transform into the certain objects of a web-architecture. As mentioned before, the client-server architecture is most common here. In order to achieve an appropriate performance, system designers are able to choose different setups. Here, the balance between the client and the server is an important topic. Regarding the third dimension, Altmaier and Kolbe mention that an interactive real-time 3D world has to be rendered on the client. This stands in contrast to fly-through movies which can be rendered on the server as well. Whereas the process of rendering for interactive real-time 3D worlds is restricted to the client, the processes of filtering and mapping can be hosted by any object of a web environment. As stated before, this should be achieved through state-of-the-art technologies like XML Web Services. In this case, one can talk about interoperable Web 3D geo-visualization (Altmaier and Kolbe, 2003). In order to implement these Web Services in standardized ways, the OGC has introduced the WFS, WCS and WMS (see Chapter 0). However, Kolbe and Altmaier mention that a fully 3D portrayal service is still missing. At the moment, only static images of 3D scenes can be served in standardized ways (Kolbe and Altmaier, 2003).

Third, the usual requirements of creating virtual worlds are important. Among others, the scene has to become closed to reality. This means, the rendered images have to be "photo-realistic". However, the overall performance has to be regarded as well. Therefore, data can not just be as detailed as possible since the rendering speed and smoothly interaction on client level is most important. Furthermore, because geographic data is very complex and usually big in terms of data size, the overall amount of rendered polygons has to be reduced as best as possible by the user agent. Here, - not to mix up with the spatial simplification or generalization of the modeled objects on database or application level – e.g. invisible polygons should not be rendered. Therefore, implementing LOD and culling algorithms are widely used. Whereas LOD is taking care of the data accuracy according to the distance between avatar and an object, culling is making sure that invisible back-faces of objects will not be rendered. Both of them are able to reduce the rendering speed significantly. Furthermore, progressive rendering techniques and dynamic content adjustments like streaming are suitable for increasing the performance (Kofler, 1998).

All these important topics should be considered in order to fulfill the goals for 3D visualization. Döllner comprises them into communication, explorative- and confirmative analysis and edition of data, concepts and models (Döllner, 2002).

2.7.1 Graphical User Interface

In order to interact and communicate with information, a graphical user interface (GUI) has to be designed and created. A GUI is located on top of the user agent. Because geographic information is usually very complex, this task is difficult to achieve. The GUI has to offer efficient access to the application logic and finally the data. Moreover, the user interface is most critical due to the fact that this is the “main gate” to the application. If a GUI is implemented poorly, an application will not be accepted by the user. In contrast to user interaction in 2D, a GUI for the third dimension has to be different. The reason for this is, because in most cases users have a poor imagination associated with multi-dimensional representations (Cöltekin, 2002). The following paragraphs are giving a brief introduction to necessary parts of a GUI.

To develop a GUI for 3D visualization on the Web, different aspects are important. First of all, as stated before, the virtual world has to be sufficient. To do so, core features of creating a 3D world are needed. The technique of visualization has to cover state-of-the-art possibilities. In the case of 3D these are reasonable modeling of physical objects, lightning and shadowing, definition of viewpoints, photo-realistic texturing. As soon as interaction has to be involved, techniques like events, linking and internal/external scripting are becoming more important. Real-time interactive navigation in a virtual world is another requirement. To explore virtual worlds a user wants to be put into this space very close. Therefore navigational characteristics like in computer games are very popular. These are for instance walkthrough, flying, panning and sliding. If the target is a singular object, rotating is another important real-time navigation attribute. In order to achieve an acceptable navigational performance, advanced characteristics of virtual worlds have to be implemented on the GUI side. As stated before, these can be Levels of Detail (LOD) or multi-resolution texturing. In order to do so, core concepts in conjunction with the application and database have to be developed. The GUI's responsibility is it to support the interaction with different data resolutions. These techniques are required to save system resources and increase performance (Kofler, 1998). Furthermore, LOD and image texturing are associated to each other. For example, Gröger *et al* proposes five LOD in order to achieve a core geo-related virtual world. Whereas the LOD 0 is textured by cartographic representations the proposed LOD 3 can be textured by images retrieved from aerial and terrestrial Photogrammetry. Other levels do not necessarily need raster-based textures. However, the number of LOD can vary individually to a certain project (Gröger *et al*, 2004).

In terms of integrated 3D visualization, Altmaier and Kolbe are specifying two approaches. First, the mosaic model is embedding neighbored cities and site models in a core regional world. Second the hierarchy scenario in which different locations contribute 3D spatial data. Therefore, different models in varying resolution are integrated to a scene (Altmaier and Kolbe, 2003).

Another sophisticated topic is the intuitive editing of 3D data. In order to provide a human readable GUI for data edition, high efforts have to be done. This is the reason why mainly common CAD or GIS software products are used as front-ends at the moment. These applications have been proven on the market for years already (Stoter and Oosterom, 2002; Zlatanova *et al*, 2002).

In terms of the third dimension, a couple of technologies are suitable for the visualization. Among others, VRML/X3D and Java3D have already proven their ability to deal with geo-information in many applications. The following paragraphs are giving a comprehensive preface on them.

2.7.2 VMRL/X3D

VRML (Virtual Reality Modeling Language) respectively its successor X3D (Extensible 3D) were introduced by the Web3D Consortium to distribute interactive virtual worlds on the web. Both are mark-up languages and standardized. Whereby X3D is fulfilling the concepts of XML. Besides X3D is specified more modular. The rendering concept is mainly based on a scene graph definition and a node structure (Web3D Consortium, 2004). VRML andX3D are accomplishing the basic concepts for a 3D GUI (Dykes *et al*, 1999). To list all the features would take too long. Concepts of a core world construction and especially the external authoring interface (EAI) grading the techniques up. By using the EAI, one can add individual functionality to virtual worlds. Developed either by scripting or higher programming languages, 3D scenes can get highly interactive. One good example is accessing a database from VRML worlds in order to retrieve new data (Zhu *et al*, 2003; Zlatanova S, 2000). Another proof that VRML/X3D is able to handle intricate tasks is shown by Diehl. His implementations show distributed virtual worlds, for instance multi-user worlds (Diehl, 2001). Furthermore, the Web3D Consortium is taking care of geographic data as well. Therefore, the GeoVRML respectively the X3D GeoSpatial working groups are founded under the Web3D Consortium (GeoVRML, 2004; X3D GeoSpatial, 2004). Geo-specific tasks like geographic coordinates, geo-elevation grid and geo-LOD are extending the usual VRML standard (Reddy *et al*, 2001). The corresponding specification of geo-related tasks in X3D is similar. However, other features like the "GeoViewpoint" are extending the standard (see X3D Specification: Part 1, 25, Geospatial component).

Realized VRML clients in combination with HTML have already proven their ability to react as GIS user agents in many examples and prototypes (see Chapter 0). However, well-known commercial implementations are not available. The most common use of VRML is within a client-side browser/plug-in implementation. Unfortunately, plug-in vendors are hesitating with shipping X3D browsers.

2.7.3 JAVA 3D

Another instrument for creating 3D world on the Web is Java 3D – an "High-Level 3D Graphics Subsystem" (Döllner, 2002). The Java3D library is a freely available API for developing Virtual Worlds in Java (Sun, 2004). Therefore Java3D classes can be used by Java Applets within HTML pages. Java3D's functionality is almost the same than VRML/X3D are providing. Savarese is introducing them briefly (Savarese, 2003). One big advantage compared to plug-in based solutions is that developers have more control about rendering and user interaction. Another is the transformability. Compiled Java3D classes can either be used as standalone application or applet. In contrast to the mark-up languages of VRML or X3D, Java3D requires much more programming knowledge (Diehl, 2001). This is probably one reason, why only a few solutions have been realized using Java3D. One example for implementing Java3D within a geo-related application is the DEMViewer by Taddei (Taddei, 2003). Another Jave3D based implementation is the GeoServNet (GSN) 3D Client introduced in Kolodziej (Kolodziej, 2003).

CHAPTER 3

METHODOLOGY

3.1 Introduction

In this chapter, the research life cycle and the operational framework with the methodology for conducting this research are presented. The aim is to develop a simple, complete and open methodology to develop some useful techniques that would improve 3D Spatial Modeling. For the development purpose, a standalone 3D building model has been designed and the spatial data of this 3D building is installed on the 3D databases management system.

This study describes a methodology based on widely available standard tools to build an interactive and highly realistic 3D building model where the display is based on 3D spatial objects in the environment extracted from the 3D spatial database. This objects data is obtained by the use of real-time objects creation algorithms technique. The resulting objects act according to the given behavior, which are blended within the objects. The objects responses are then transmitted to other users. The 3D building application that is used for real-time rendering and visualization is based on standard language VRML. Using this process one can create objects, attach responses with objects and navigate real-time Virtual Environments (VE) on any operating. The techniques for VE's are modeled for maximum precision and accuracy.

3.2 Data Sources and Instrumentations

Figure 3.1 shows the summary of this research in form of flow chart. In the figure, all equipments used have been mentioned in this research in terms of software's and hardware's. Tools such as Oracle 10g as a database management system with spatial cartridge, VRML/X3D, JAVA, XHTML and programming software e.g. Autodesk Map 3D 2005 used for research work are also stated. Research life cycle and Operational framework in the flowing sections present the detail of all phases that have been used for this research including the details of problem formulation, system development, implementation, integration of different modules and testing of designed system. Furthermore, after performing all these tasks the most important task of this research is to document this research. Finally, in achievements, some enhancements of previous work and designing of new data techniques developed during this research have been presented.

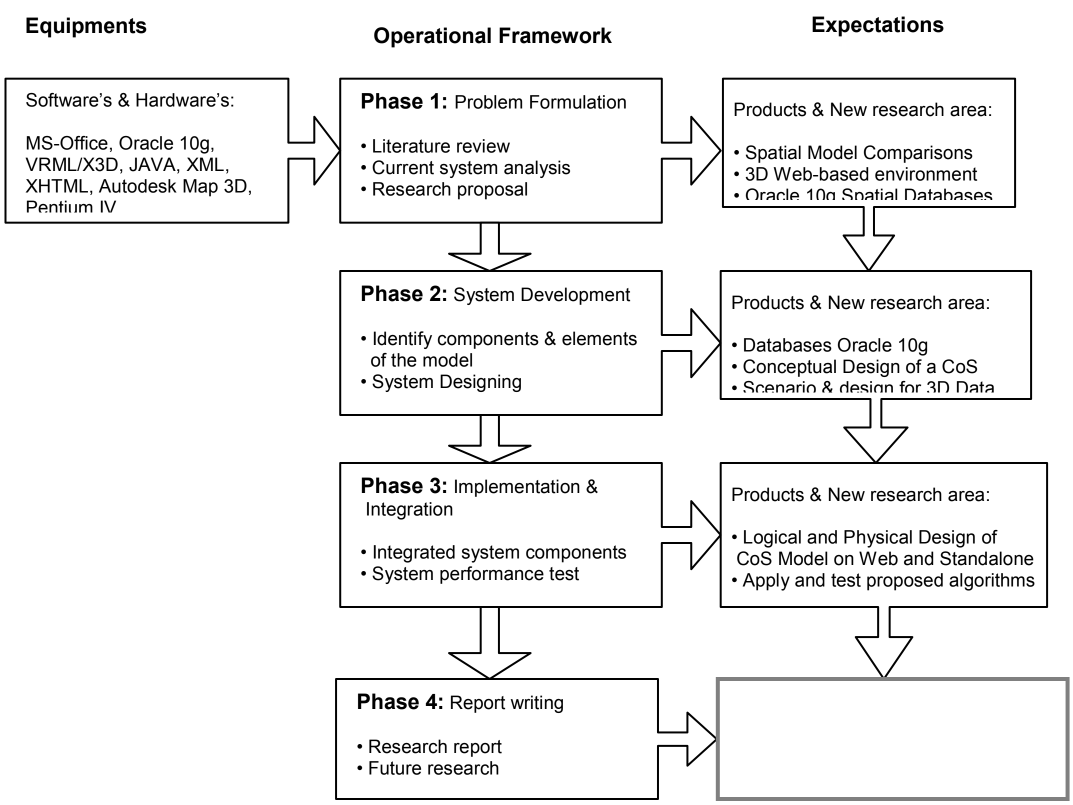


Figure 3.1: Data Sources and Instrumentation Used in this Research

3.3 Research Life Cycle

This research is a significant effort towards the betterment of 3D spatial model and 3D web-based GIS System. To conduct this study successfully, a research life cycle plan (Figure 3.2) has been designed, applying UML (Unified Modelling Language) sequence diagram.

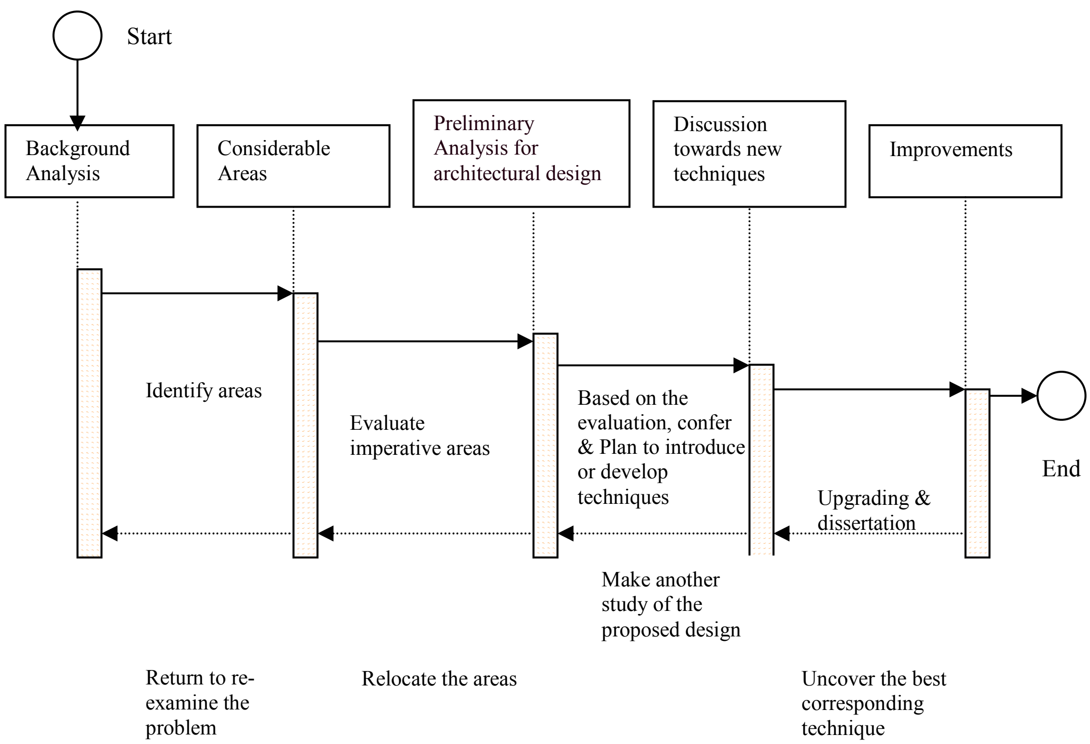


Figure 3.2: Research Life Cycle Sequence Diagram

The research life cycle works as an upper-layer over Operational *Framework* (section 3.4) and has guided us to have a Conceptual Outline of our on going effort from start to end. On the basis of research life cycle plan, an operational framework (Figure 3.4) comes into existence to further intricate the research by dividing it into four main phases. The research life cycle plan is a milestone to know, understand and manage several aspects of conducting this research so as to provide connectivity among different segments while working in a consistent and coherent way. The detailed description of the above Figure 3.2 is provided below.

3.3.1 Background analysis

3D spatial data modeling and its visualization is an enormous research area. Today, a growing number of systems, models and techniques are being fashioned, which facilitates the 3D community up-to some level. With the quick growth in 3D systems, some significant issues have raised which create complexities for users. This phase, however, started to explore and investigate those issues in detail that need attention to be addressed. In chapter 2, popular 3D spatial models and 3D web-based systems and their approaches have been analyzed. From the analysis of those approaches we get solid idea about the intensity of the challenges 3D web-based GIS systems are facing, such as relationships among 3D complex object and their analysis.

3.3.2 Areas to be evaluated for considerations

Based on above analysis and former work this phase proposes to highlight the areas that need consideration or re-consideration. In our case of spatial relationships, for instance, it was identified that focus should be put for the analysis, to reduce the data redundancy and also data size of the 3D data set that are involved in a complex 3D environment at a given time. It was felt that previous models needs some improvement in terms of handling spatial relationships, data consistency and data redundancy and it also effect on the storage size of a 3D object.

3.3.3 Preliminary analysis to architectural design based on the previous evaluation

This segment focuses on initial investigation about the structure of the techniques to be improved or extended on a conceptual level. As this research emphasis to reduce cost of storage to improve 3D spatial model by egenhoffer i.e. 9i Model, so it addresses the issue from lower to upper end i.e. from objects creation, their relationships and data storage.

3.3.4 Discussion towards new techniques

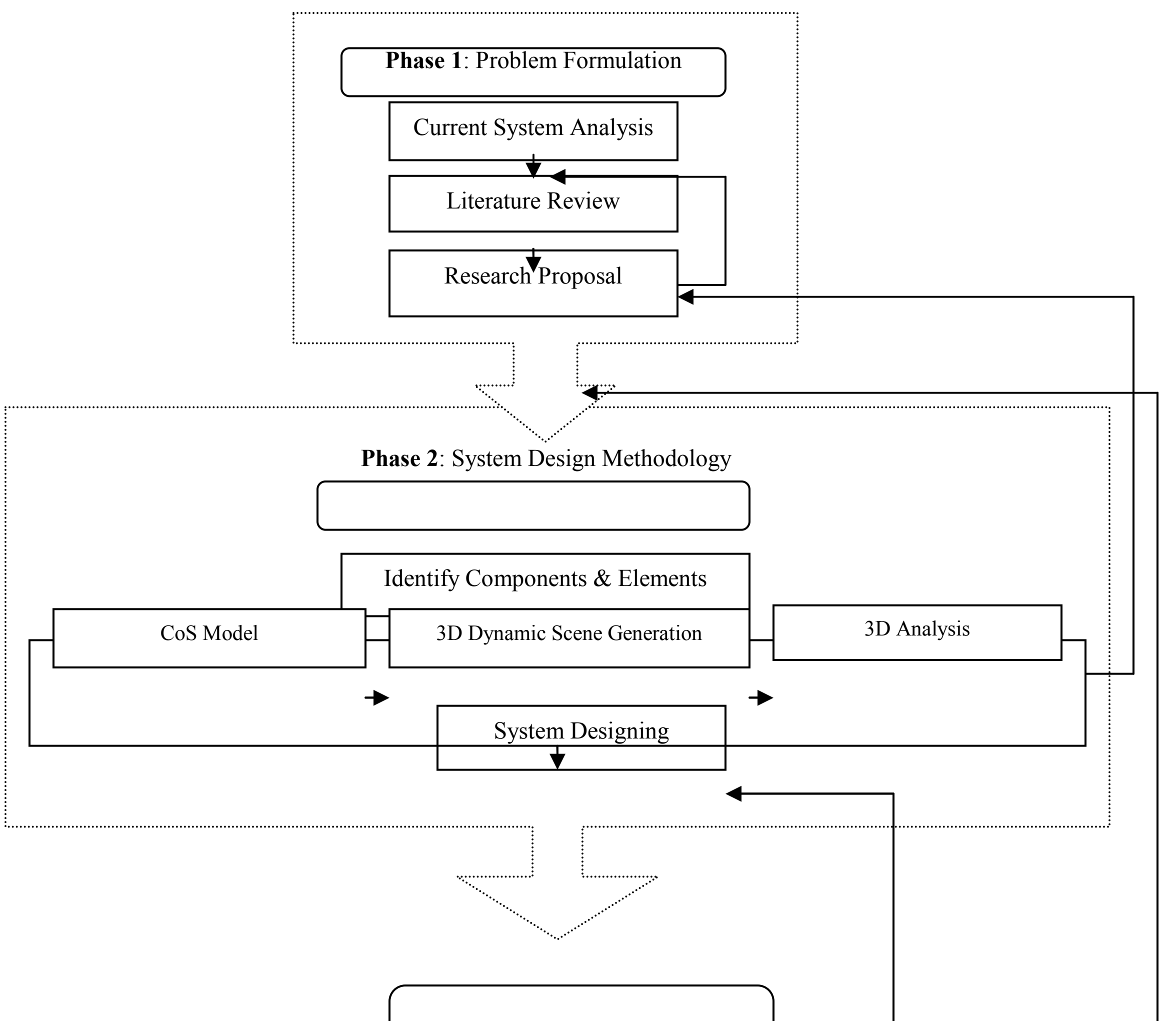
In order to handle the 3D spatial relationships and the analysis on the web, some well-designed approaches are needed. This is where; we need to determine what methodology to be used for developing the effective techniques based on the findings. So, it has been decided to improve the specific algorithms systematically that create objects, provide analysis facilities of 3D spatial object on the web.

3.3.5 Improvement

In this part, the proposed algorithms or techniques are implemented to evaluate the requirements and quality of performance. This step shows whether the developed approach and its results are predisposed for usage or still needs some more enhancements in research work.

3.4 Operational Framework

The operational framework of the research mainly consists of problem formulation phase, system design phase, system development phase and finally the report-writing phase. In the very first phase of the proposed operational framework Figure 3.3 describes the problem formulation phase, which includes literature review, current system analysis and presenting research proposal, details of this phase is presented in (section 3.4.1). In the system design methodology phase (section 3.4.2), 3D GIS considerations, the significant components and approaches like objects creation, data access from oracle and are described. The System development and integration phase (section3.4.3) presents the construction and development of the proposed problem. Finally section 3.5 discusses the way, how the report-writing phase goes together with overall research. Figure 3.3 illustrates the operational framework of this research and is used as basis to draw the methodology of this research.



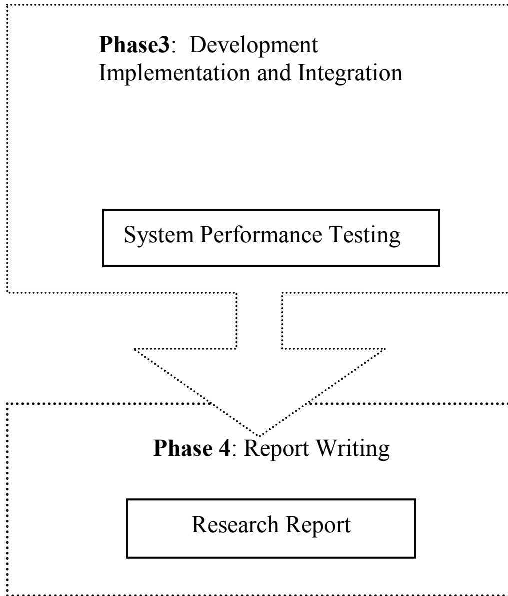


Figure 3.3: Research Operational Framework

3.4.1 Problem Formulation (Phase 1)

The problem formulation phase is considered a leading step to conducting this study. Title selection, setting up basic scope and ranging the core objectives of the study are being determined at this stage. In addition to this, literature review is one of the important methods that contribute many ideas in developing and designing all kinds of study i.e. journal, conference papers, articles, books and Internet libraries are the sources of this literature study. Information collected from literature review range from the basics of the system development, concept, method, techniques and current trends of development in which all of these can be used as references and sources for new innovative ideas and developing the proposed system. The literature review conducted for this study is already presented in previous chapter but this is a continuously running process, and this process is parallel in order to achieve optimum results.

3.4.2 System Design Methodology (Phase 2)

In order to develop a well-designed system, a systematic approach is needed. As been described earlier, this is where we need to determine which system development methodology to use. In the current research case, the whole system can be divided into three major parts as: CoS model, dynamic scene generation (objects creation, interaction/relationships) and 3D Analysis. After that, based on the previously discussed components and elements, system design with the decision of best topological architecture is constructed for the development and implementation of 3D web-based GIS system. First phase (Figure 3.2) ensures whether requirements regarding 3D web-based GIS system considerations are clear measurable, achievable, and complete.

3.4.3 Identify components and elements

This section summarizes some focused areas that are realized as important parts of the 3D web-based GIS system while discussing about Analysis. The framework has composed the component and element for a 3D web-based GIS system into main step, categorized as follows:

(a) Objects design building (ODB)

As we have mentioned earlier somewhere else that our proposed technique includes various operational parameters to establish a real-time creation of object on the web through database management system for multi-users. One of them is objects creation. One of the main criteria's to measure the stability of 3D web-based GIS system is dynamic objects creation. A virtual environment deserves significance if it creates and renders the virtual scene quickly.

3.4.4 System Designing

System designing phase helps to understand the proposed development of major objectives. Based on the identification of the main components and elements, the system design phase comes to play role to give a formal shape to this research. This phase is focuses to see the possibilities to group the activities of different processes, which are functioning independently so as to acquire the maximum success and performance. This phase serves as a concurrent-engineer in this research, as this integrates and collaborates multi-functional distinct sub-systems i.e. objects creation, interaction/relationship into one 3D web-based GIS system. Development, Integration and Implementation (Phase 3)

Several system components that are developed in different phases and at different time, need to be integrated so as to provide a coherent output. For quality assurance purposes, this includes the investigation of system performance as well. The following sections throw light on this note.

3.4.4.1 Integrating system components and Implementation

The described components and elements are functionally integrated at this phase. This becomes an important stage as all the efforts made and components (algorithms) evolved earlier are grouped together to enable the system to work in an organized way. This is essential because for many resources used, the resultant product quality is used to measure the assessment of performance and success.

In the on going research, system is be implemented for real time interactivity for multi-users. Hardware platforms employed by the researcher is a computer machine Pentium IV processor with 1GB of memory and running Windows XP Pro. The system is connected through a LAN network.

The results of the study have been implemented by means of VRML/X3D and Oracle 10g Spatial Cartridge that processes the extraction to derive and interact with the geometry of the objects to display it into some output. The CoS model has been implemented and developed by using Oracle Spatial database.

3.4.4.2 System Performance Testing

The performance of the system can be tested by multi-users connected on the web. The architecture for objects creation, for relationships and for data preparation to send on web can be applied and tested for fundamental experiments. The analysis is provided and to recognize each object that which is in the environment, and who is interacting with the virtual world. In this way, efficient responses of into web-based 3D GIS system can be verified when more than one user performs analysis on the environment.

3.4.5 Report Writing (Phase 4)

Report writing is the last part of the study, which does all the writing up including the documentation of the system and user manual. In this research thesis objects creation, relationship/interaction to send on web, which have been enhanced and implemented for web-based 3D GIS system, are reported. This is the phase where all results and discussion regarding the developed techniques with the description of future work are presented and concluded.

3.5 Summary

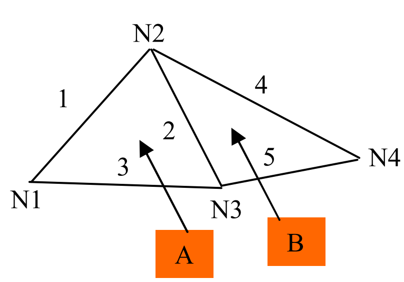
To meet the mentioned challenges, the proposed research life cycle plan as well as operational framework play a vital role to guide us for the research methodology, development flow and results analysis of our web-based 3D GIS system. The proposed framework has made it easy to investigate some important issues, to construct design for analysis of those issues and to develop virtual environment for web with some experience of analysis, dynamic objects creation and 3D analysis. By conducting this research, some significant design guidelines have been proposed, which would help the researchers to understand different scenarios of real-time interactive visual web-based 3D GIS system.

CHAPTER 4

CONDENSED SPATIAL (CoS) MODEL

4.1 INTRODUCTION

In GIS, the vector data model that used for geographic phenomena may be represented by geometric entities (primitives) like points, lines, polygons, and volumetric solid object. With the integrated Geo-DBMS module (such as Oracle Spatial, 2002), geometric objects can be stored together with topological information. These vector data models that include the description of topology, as well as the location of the spatial entities will be stored. In general, topology in GIS requires a data structure, where common boundary between two adjacent areas is stored as a single line, simplifying the map maintenance. On the other hand, geometric entities require full data insertion of any object. For instance, two triangles that share a common boundary stores topological information of six lines. However, in geometric condition, they will be stored twice of four nodes. Figure 3.1 gives the difference between topological and geometric data storage in Geo-DBMS.



TOPOLOGICAL DATA		GEOMETRIC DATA	
FaceID	LineID	FaceID	NodeID
---------	---------	---------	-------------
A	1-**2**-3	A	N1-N2-N3-N1
B	4-5-**2**	B	N2-N4-N3-N2

Note: 1). **2** represents the common boundary
2). Line **2** represents node N2 to N3

Figure 3.1: Comparison between topological and geometric data

Beside reducing data storage, Geo-DBMS allows multi-user control on shared data and crash recovery, automatic locks of single objects while using database transactions, advanced database protocol mechanisms to prevent the loss of data, data security, data integrity and operations that comfortably retrieve, insert and update data (Bruenig and Zlatanova (2004), Patel *et. al* (1997)). In this paper, capabilities of topology in handling spatial datasets in GIS will be discussed. It involves spatial primitives that deal with spatial data recognitions (semantics) and their relationships among objects. A new framework for representing spatial model named *Condensed Spatial* (CoS) model is introduced. The model implements topological mechanism for object semantic and relationship that able to represent the real world, i.e. node, line, face, solid3D. This paper is organized in the following order. First, the properties of each primitive will be described. Then, the main study of topology that involves primitive and feature objects' definitions will be discussed extensively. Later on, the visualization of CoS model will be highlighted, i.e. integration between Oracle database and AutoDesk Map 3D module will be mentioned. Finally, the paper concludes with experiment and result discussions.

4.2 GEOMETRIC PROPERTIES

4.2.1 Node

Node is defined as single coordinate triplet represented by (x, y, z) in three-dimensional space ($\mathbf{R}^3$). It appears as 0D object in 3D Euclidean space. It is used to represent objects that are best described as shape- and sizeless, single-locality features. The location of each different node must be unique. The interior of a point is the empty set, denoted by P^o. For any cases, interior of point will not be related to any kind of primitives due to the intersection results is an empty set. The boundary of a point, denoted by ∂P, is the point by itself. Finally, the exterior of a point is denoted by P^-.

4.2.2 Line

Line is defined as series of nodes connecting together with an appropriate sequence. It is a one-dimensional spatial entity that defines a path through 2D or 3D space. The geometry of line is defined by an ordered collection of two or more distinct coordinate tuples, as shown in Figure 3.2. The orientation of a line is defined by the ordering of its coordinate tuples.

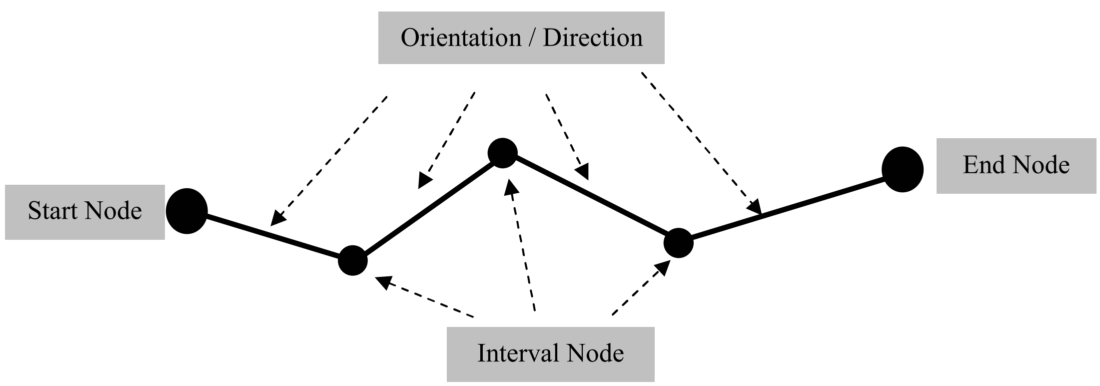


Figure 3.2: Geometry of a line

A line is used to represent one-dimensional objects such as road, railways, canals, rivers, or even power lines in 3D Euclidean space. A line connecting two nodes forms an arc. However, two end nodes connecting a vertex or more vertices define a line. The straight parts of a line (or an arc) between two consecutive vertices are called line segments. Collections of (connected) lines may represent as network in real applications. These end points are defined as the border of a line, denoted by ∂l. The interior of line is defined as the line segment itself, denoted by l^o. The exterior of line is denoted by l^-.

4.2.3 Face

Face is defined as a planar areal object. This areal feature is determined by series of lines with an appropriate connected sequence, forming a closed boundary. It appears as 2D object in 3D Euclidean space. A face connects three points forms a triangle, otherwise, with at least four points forms a polygon. In most cases, the coordinate tuples of the bounding edges of a face are not necessarily coplanar. Any entity nodes that are contained within the face also help to define its three-dimensional shape. The starting and ending points are the same point that defined as the border of polygon (see Figure 3.3), denoted by ∂Po.

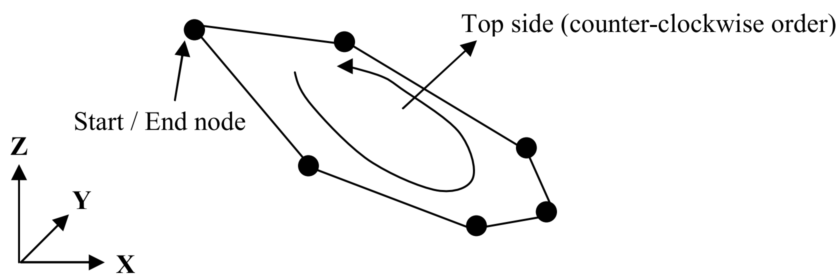


Figure 3.3: Geometry of a face

In three-dimensional space, a face shares only between two solid objects. This implies a face has two sides. The orientation of a face is defined by the order of the edges that make up its outer boundary. The "top" side of a face is the side for which the outer boundary is defined in counter-clockwise order; otherwise the inner boundary defines in clockwise order. In order to support vertical and overhanging surfaces, the orientation of the "top" side of a face must be capable of being defined to be an arbitrary direction, not necessarily parallel to the positive Z-axis. Therefore, a face must have an explicit "up" vector (refer to Figure 3.3). Either a simple face or face with holes, the properties of face will not affect any topological relationship between other objects. The different between these two kinds of face is the face with hole consists of two or more borders, whereas the simple face only remains one border. The interior of a face is defined as the area within its boundary, denoted by Po^o. The exterior of face is denoted by P^-.

4.2.4 Solid3D

A solid3D is defined as indexed set of faces (polygons) joining together that forms a volumetric object. It appears as 3D object in 3D Euclidean space. A solid3D may be topologically linked to nodes, edges, and faces (see Figure 3.4). These nodes, lines, and faces that form a solid3D are defined as the border of solid, denoted by ∂So. In the PR model, either a simple 3D object or solid object with holes, the properties of polygon will not affect the topological relationship between other objects. The different between these two kinds of solid3D is the one with hole consists of two or more borders, whereas the simple solid3D only remains one border. The interior of solid3D is defined as the closure of all borders, denoted by So^o. The exterior of solid3D is denoted by So^-.

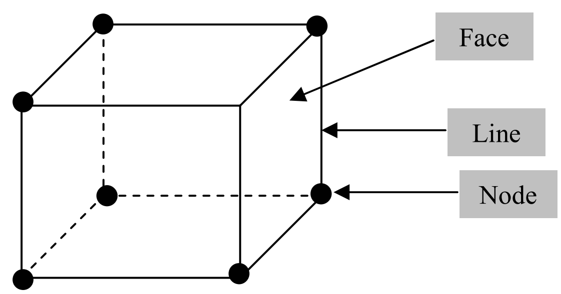


Figure 3.4: Geometry of a solid3D

4.3 CONDENSED SPATIAL (CoS) MODEL

In this section, the definition of a new spatial model will be given. It will be referred to as Condensed Spatial (CoS) model. In order to produce a model that implements only the most essential primitives, arc and surface are not used to construct objects. Due to the arcs store multiple nodes information within Geo-DBMS database, lines are used to reduce datasets storage and improve the abilities of data retrieval and spatial query. Besides, a surface created from multiple faces also omitted from this model. This is because a single but planar face may give a common boundary either for two solid objects, or topological space and an object. On the other hand, a surface that created by faces may or may not represent a common boundary between two solid objects. Furthermore, topological relationship that implements surface will become more complex, i.e. more relationship will be involved within node-surface, line-surface, surface-surface, or even surface-solid (Chen *et. al*, 2005). Therefore, three primitives are given, i.e. node, line, and face.

4.3.1 Definition of primitive objects

In topological space, T,

Definition 1: The *node* denoted by N_i, where i is the *unique index* of a node, with the following property:

a). N_i is represented by coordinate triplet (x, y, z) that denotes a location in $\mathbf{R}^3$,

b). Two nodes could not construct same feature object, i.e. two traffic sighs are always *disjoint*,

c). The *interior* of a node, denoted by $N°$, is the empty set. The *boundary* of the node, denoted by ∂N,
is the node by itself. The *exterior* of a node, denoted by N^- is everything but not the node itself.

Definition 2: The set of all the N_i nodes in a topological space, T, is denoted by AN have the following properties:

a). The intersection of all the nodes (in topological space, T) is the non-empty set, i.e. $\{AN_i \cap AN_j\}^{\mathrm{T}} \neq \emptyset$, due to nodes may be part of some feature object, i.e. points, or lines,

b). Two nodes, N_i and N_j in $\mathbf{R}^3$ are connected, if and only if there is a straight line linking them, otherwise they are disconnected. The straight line connecting two nodes will be referred to as *line segment* (or *arc*) in this context.

c). There are nodes, which does not constitute point(s). However, these nodes must be part of line(s).

Definition 3: A *line* denoted by L, is an indexed set of x ordered nodes $N_i, \subset AN_i$,

$$L_i = \bigcup_{i=1}^{x} N_i$$

where x is the total number of node N_i, and i is the index of the node specifying the current order in a line, L_i, with the following properties:

a). $x \geq 2$, (if $x = 2$, L_i represents an arc or line segment),

b). A line segment fulfils only the linear equation $y = mx + c$, where $m = (y_{i+1} - y_i) / (x_{i+1} - x_i)$, c denotes the y-intercept, and (x,y) denotes any node from that line segment.

c). In the set of nodes cannot exist two equal nodes, i.e. the intersection of the nodes (within an ordered set of a line, L) is the empty set, i.e. $\{N_i \cap N_j\}^L = \emptyset$.

d). The *interior* of a line, denoted by $L°$, is a set of line segments. The *boundary* of a line, denoted by ∂L, is the starting and ending node. The *exterior* of a line, denoted by L^- is everything but not the line itself.

Definition 4: A set of lines in a topological space, T, is denoted by AL have the following properties:

a). The intersection of the lines (in topological space, T) is the non-empty set, i.e. $\{AL_i \cap AL_j\}^T = \emptyset$, b). The intersection of a set of arcs (line segment) of a line, *Ar*, is empty set, $\{Ar \cap Ar\}^L = \varnothing$.

Definition 5: A *face* denoted by F is an indexed set of x ordered lines, $L_i \subset AL_i$,

$$F_i = \bigcup_{i=1}^{x} L_i$$

where x is the total number of line L_i, and i is the index of the line specifying the current order in a face, F_i, with the following properties:

a). $3 \le x \le n$, (if $x = 3$, F_i represents a triangle), where n is a finite number,

b). Two equal lines will not exist within a set of a face, F_i

c). The intersection of the lines (in an ordered set of a face, F) is the empty set, i.e. $\{L_i \cap L_j\}^F = \varnothing$.

d). The *interior* of a face, denoted by F°, is an area inside the lines. The *boundary* of a face, denoted by ∂F, is the set of lines. The *exterior* of a face, denoted by F^- is everything but not the line and face themselves.

Definition 6: A set of faces in a topological space, T, is denoted by AF have the following properties:

a). The intersection of the faces (in topological space, T) is the non-empty set, i.e. $\{AF_i \cap AF_j\}^T = \varnothing$,

b). A face shares a common boundary not more than two solid objects.

4.3.2 Definition of Feature objects

The geometry of each spatial object can be associated with four abstractions of geometric objects, i.e. *point*, *line*, *face* and *solid*. A *point* is a type of spatial object that does not have shape or size but position is the topological space. A *line* is a type of a spatial object that has length and position. A *face* is a type of spatial object that has position and area. A *solid* is a type of spatial object that has a position and a volume. In order to construct a spatial object, a new topological framework that involves the design of spatial model, named as Condensed Spatial (CoS) model is developed. Figure 3.5 denotes the conceptual design of CoS model.

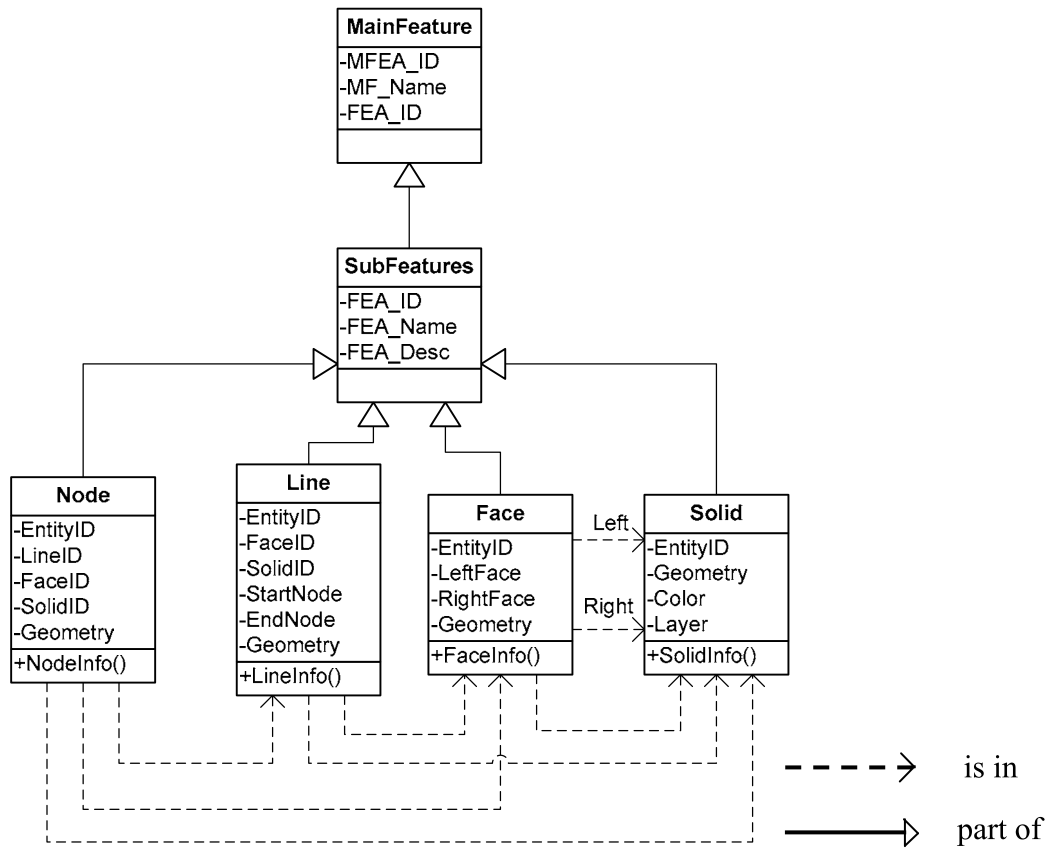


Figure 3.5: The conceptual design of Condense Spatial (CoS) Model

The three primitive objects are used to compose the four feature objects, i.e. *point*, *line*, *face* and *solid object*. However, the feature objects are constructed using these rules as follows: nodes construct points and lines; lines construct faces; faces construct solid objects. This section gives formal definitions and specifies the properties of feature objects. The possible relations between primitive and feature object and the rules to construct topology, which can be derived from the definitions, are discussed.

Definition 7: A *point* denoted by P_i is an indexed set of nodes N_i, if and only if N_i is not part of other feature object, i.e. line, face and solid object.

$$P_i = N_i = \{x, y, z\}_i$$

where i is the index set of the nodes.

The topological primitives of a point are:

a). The *interior* of P, denoted by P° is empty, i.e. $P_i^\circ = N_i^\circ = \varnothing$,

b). The *boundary* of P, denoted by ∂P, is the node itself, i.e. $\partial P_i = \partial N_i \neq \varnothing$,

c). The *exterior* of P, denoted by P^-, is everything but not the node itself, i.e. $P^- \neq \varnothing$.

Definition 8: In a topological space, T, the set of all the points $\{P_i\} \subset T$ has the following properties:

a). The intersection of all the points is equal to the empty set, i.e. $P_i \cap P_j = \varnothing$, i.e. only one feature object of a point in the same location. This means that the existence of two *equal* points is not possible in this context. For example, two traffic signs are located in different locations.

b). For some reasons that point $\{P_i\} \subset T$, nodes $\{N_i\} \subset T$ does not constitute $\{P_i\}$, it must constitute line(s). This means $\{N_i\} \in L_j$ and point P_i may *meets* the line L_j. Otherwise the point P_i and the line L_j are *disjoint*.

According to the definitions above, a common node may constitute a point and line. Points can coincide with the boundary of a line.

Definition 9: A *line* denoted by L, is an indexed set of x ordered nodes $N_i, \subset AN_i$,

$$L_i = \bigcup_{i=1}^{x} N_i = \{N_i, N_{i+1}, \ldots. N_n\}$$

where x is the index set of nodes N_i, n denotes the total number of nodes, and i s the index of the node specifying the current order in a line, L_i.

The topological primitives of a line are:

a). The *interior* of L, denoted by $L°$, represents its line segment between first and end nodes, is non-empty, i.e. $L_i^{o} \neq \varnothing$,

b). The *boundary* of L, denoted by ∂L, is the first and end nodes, is non-empty, i.e. $\partial P_i = \partial N_i \neq \varnothing$,

c). The *exterior* of L, denoted by L^{-}, is everything but not the line itself, i.e. $P^{-} \neq \varnothing$.

Figure 3.6 denotes the topological properties of line.

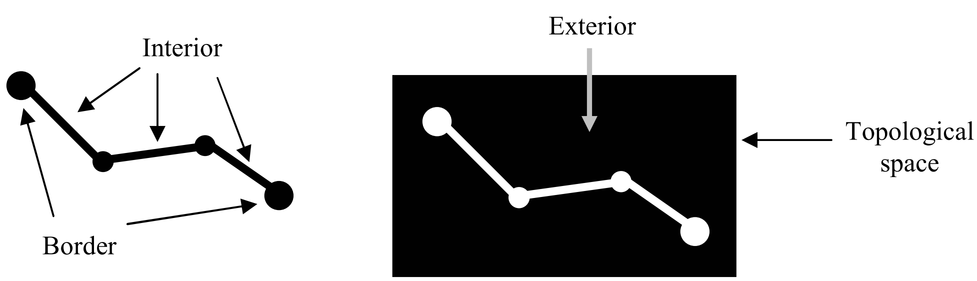


Figure 3.6: Topological properties of line

Definition 10: In a topological space, T, the set of all the lines $\{L_i\} \subset T$ has the following properties:

a). There is one set of nodes, which defining a line, denoted by first node, N_1 and last node N_n, and interval nodes, $\{N_i\}$.

b). The line is *closed* if $N_1 = N_n$.

c). The intersection among nodes ($N_i \in L_i$) is the empty set, i.e. $N_i \cap N_j = \varnothing$.

d). The intersection of all the *lines* is not equal to the empty set, i.e. $L_i \cap L_j \neq \varnothing$.

e). A line may be a feature object, e.g. railway route, or transmission line; otherwise it is a part of face, $L_i \in F_i$.

Definition 11: A *face* denoted by F_k is an indexed set of x lines L_j, $x \geq 3$ (if $x = 3$ the *face* is called a *triangle*),

$$F_k = \bigcup_{j=1}^{x} L_j = \{L_j, L_{j+1}, \ldots L_n\}$$

where k is the index set of a face with the following topological primitives:

a). The *interior* of F, denoted by F°, represents the area within boundary, is non-empty, i.e. $F_i^{\circ} \neq \varnothing$,

b). The *boundary* of F, denoted by ∂F, is the line segments and nodes, is non-empty, i.e. $\partial F_i = \{N_i \cap L_i\} \neq \varnothing$,

c). The *exterior* of F, denoted by F^{-}, is everything but not the face itself, i.e. $F^{-} \neq \varnothing$.

Figure 3.7: Topological properties of face

Definition 12: In a topological space, T, the set of all the faces $\{F_i\} \subset T$ has the following properties:

a). There is one set of lines, which defining a face, denoted by $\{L_i\} = \{L_i, L_{i+1}, \ldots L_n\}$ are closed if both L_i and L_n are connected.

b). The intersection among lines ($L \in F_i$) is the empty set, i.e. $L_i \cap L_j = \emptyset$. This means lines are not intersecting among themselves from a set of face (see Figure 3.8), otherwise this face will be split into faces.

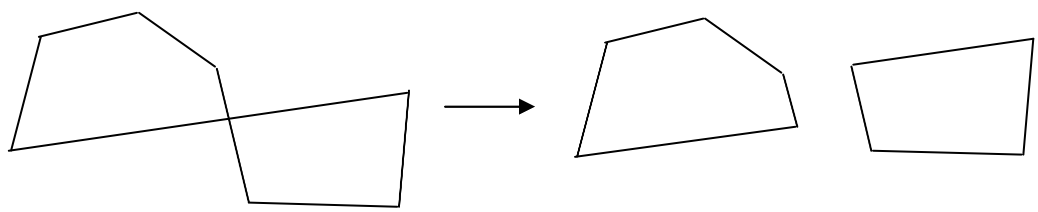


Figure 3.8: Face separation (follows the rules, $L_i \cap L_j = \emptyset$, where $L \in F_i$)

c). The intersection of all the faces is not equal to the empty set, i.e. $F_i \cap F_j \neq \emptyset$.

d). A face may be a feature object, e.g. land area; otherwise it is a part of solid object, $F_i \in So_i$.

Definition 13: A *solid* object denoted by So_k is an indexed set of x faces F_j, $4 \leq x \leq m$ (if $x=4$ the *solid* is called a *tetrahedron*),

$$So_j = \bigcup_{i=1}^{x} F_i = \{F_i, F_{i+1}, \ldots F_n\}$$

where j is the index set of a *solid* with the following topological primitives:

a). The *interior* of *So*, denoted by So°, represents the volumetric object, is non-empty, i.e. $So_i^{\circ} \neq \emptyset$,

b). The *boundary* of *So*, denoted by ∂So, is the nodes, line segments and faces, is non-empty, i.e. $\partial F_i = \{N_i \cap L_i \cap F_i\} \neq \emptyset$,

c). The *exterior* of *So*, denoted by So^-, is everything but not the solid itself, i.e. $So^- \neq \emptyset$.

Definition 14: In a topological space, T, the set of all the faces $\{F_i\} \subset T$ has the following properties:

a). There is one set of faces, which defining a solid object, is denoted by $\{F_i\} = \{F_i, F_{i+1}, \ldots F_n\}$.

b). The intersection among lines ($F_i \in So_i$) is the empty set, i.e. $F_i \cap F_j = \varnothing$. This means faces are not intersecting among themselves from a solid object, otherwise solid object will be split.

c). The intersection of all the solid objects (in T) is not equal to the empty set, i.e. $So_i \cap So_j \neq \varnothing$.

d). A solid object must be a feature object, e.g. room in a building.

e). Two solid objects are *joint* if they share a common boundary; node(s), line(s), or face(s), otherwise they are *disjoint*.

The 14 definitions given above complete the description of the Condensed Spatial (CoS) model. The model consists of three primitive objects, (*nodes*, *lines*, and *faces*) and four geometric objects (*point*, *line*, *face* and *solid object*). Nodes constitute points and lines. Lines constitute faces and faces constitute solid objects. The definitions specify the permitted shape of the constructive and geometric objects, as well as establish rules to compose geometric object from primitive objects. Each geometric object has specified *topological primitives* as well. All the statements are presented by notation of set theory under the assumption that the objects are embedded in Euclidean space.

4.4 Architecture for Web-Based 3D GIS

4.4.1 VRML and X3D

In 1994 the Web3D Consortium launched VRML (Virtual Modeling Language), which became an international ISO standard in 1997. The basis for the development of

VRML was to have a simple exchange format for 3D information. This format is based on the most used semantics of modern 3D applications: hierarchical transformations, illumination models, viewpoints, geometry, fog, animation, material characteristics and texture.

The development of VRML has stopped since the Web3D Consortium started to work on a XML version of VRML, in order to integrate with other web technologies and tools: X3D (eXtensible 3D). The specifications of X3D have only recently become available May, 2003. In our research we use both X3D and VRML to visualize 3D geo-information. The data structure of an X3D document is very much comparable to the data structure of a VRML file. So as far as the underlying data model is concerned, X3D must be seen as a subset of VRML (Web3D, 2003). The difference lies in the notation (the syntax) used. While VRML is text, with accolades for structuring, X3D is coded in XML, with 'tags' for structuring. This is a major advantage for on-the-fly retrieval, because of the ease of use of XML in Internet applications.

4.4.2 PHP and VRML/X3D Integration

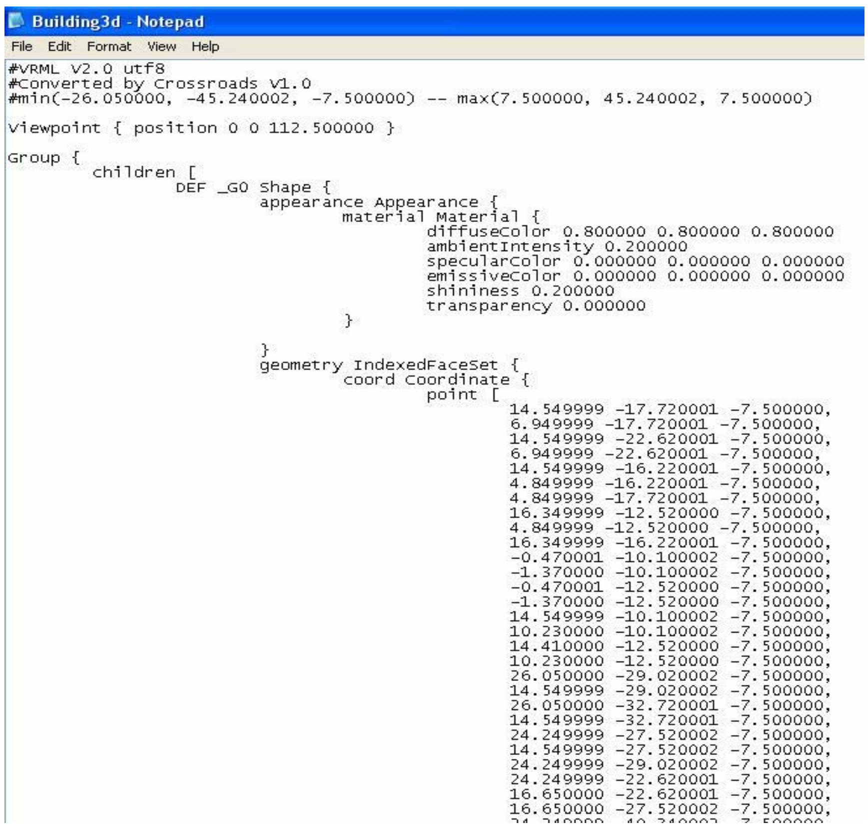
PHP is becoming very popular language for creating dynamic websites, particularly for generating them from databases. However, 2D GIS are outdated, aren't they? What we want is a way to harness the power of PHP to create database-driven VRML worlds for new 3D GIS system.

PHP can output any text information; the only thing that identifies the file type is the MIME type that gets sent by the web server before the file itself. Fortunately, in PHP, one has to take control of this very information. This requires sending the VRML MIME type (“model/vrml”), and then writing the appropriate VRML nodes.
The server strips all PHP code when sending a response. So, on lines where only JSP code is present, the server simply sends blank lines back to the browser.

it's necessary to include both PHP and VRML headers, and the content type must be changed before we set the VRML header, , so the final result looks like this:

```
<?php
  header ("Content-type: model/vrml");
  echo "#VRML V2.0 utf8\n";
?>
```

In this research experimentation objective is to display a 3d Building in VRML code using database. But in the start, it has been tested with the same type of data as used in Oracle Spatial to display a VRML file. In Figure 3.9 shown below the VRML file contains data in itself.




```
#VRML V2.0 utf8
#Converted by Crossroads v1.0
#min(-26.050000, -45.240002, -7.500000) -- max(7.500000, 45.240002, 7.500000)

Viewpoint { position 0 0 112.500000 }

Group {
        children [
                DEF _G0 Shape {
                        appearance Appearance {
                                material Material {
                                        diffuseColor 0.800000 0.800000 0.800000
                                        ambientIntensity 0.200000
                                        specularColor 0.000000 0.000000 0.000000
                                        emissiveColor 0.000000 0.000000 0.000000
                                        shininess 0.200000
                                        transparency 0.000000
                                }

                        }
                        geometry IndexedFaceSet {
                                coord Coordinate {
                                        point [
                                                14.549999 -17.720001 -7.500000,
                                                6.949999 -17.720001 -7.500000,
                                                14.549999 -22.620001 -7.500000,
                                                6.949999 -22.620001 -7.500000,
                                                14.549999 -16.220001 -7.500000,
                                                4.849999 -16.220001 -7.500000,
                                                4.849999 -17.720001 -7.500000,
                                                16.349999 -12.520000 -7.500000,
                                                4.849999 -12.520000 -7.500000,
                                                16.349999 -16.220001 -7.500000,
                                                -0.470001 -10.100002 -7.500000,
                                                -1.370000 -10.100002 -7.500000,
                                                -0.470001 -12.520000 -7.500000,
                                                -1.370000 -12.520000 -7.500000,
                                                14.549999 -10.100002 -7.500000,
                                                10.230000 -10.100002 -7.500000,
                                                14.410000 -12.520000 -7.500000,
                                                10.230000 -12.520000 -7.500000,
                                                26.050000 -29.020002 -7.500000,
                                                14.549999 -29.020002 -7.500000,
                                                26.050000 -32.720001 -7.500000,
                                                14.549999 -32.720001 -7.500000,
                                                24.249999 -27.520002 -7.500000,
                                                14.549999 -27.520002 -7.500000,
                                                24.249999 -29.020002 -7.500000,
                                                24.249999 -22.620001 -7.500000,
                                                16.650000 -22.620001 -7.500000,
                                                16.650000 -27.520002 -7.500000,
```

Figure 3.9: 3D Building's VRML File

PHP scripting is added to this shape, which lets us use dynamic data to change the sphere's position in space (translation X Y Z), its color (*diffuseColor R G B*), and its radius.

4.5 Prototype and Process details

The basic idea of the prototype is to organize 3D geo-objects in a DBMS and to query them via an Internet browser. Geo-objects contain both spatial and non-spatial (administrative) information. The spatial information can be visualized after conversion into VRML or X3D and the non-spatial attribute information can be presented in (dynamic) HTML pages.

Figure 3.10 shows a standard request process—the page is requested via a browser. The request calls the designated PHP, which interacts with a database. The model given below explains the whole prototype process.

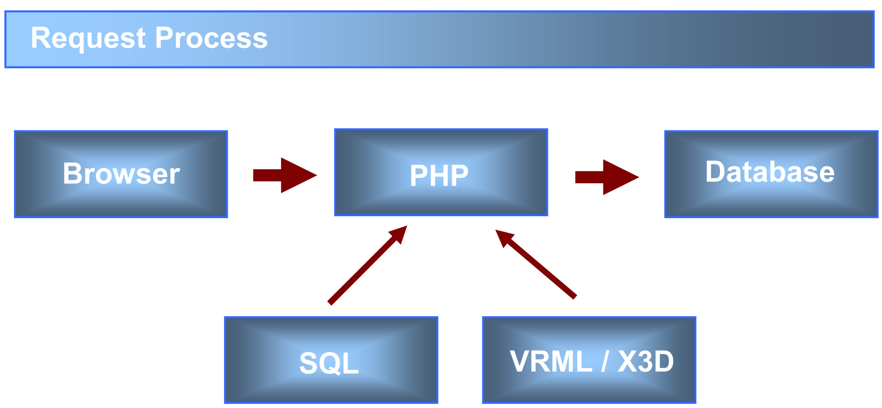
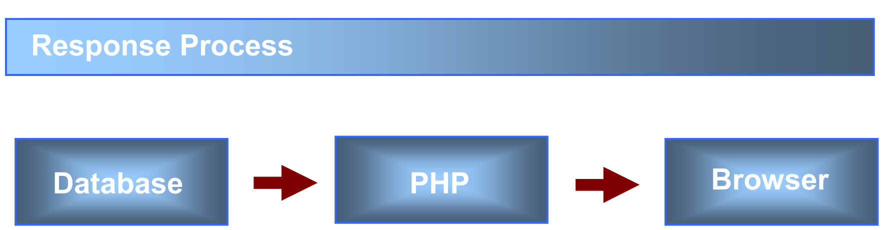


Figure 3.10: Request Process

After receiving the response, the system follows the flow shown in Figure 3.11. The database sends the requested data to the PHP, which formats the data and sends the response to the requesting browser. In our case study, the data is returned to the PHP, which generates a VRML scene using the data from the database.



On a client request a connection is made to the DBMS and the spatial information of interest is selected from the DBMS and converted into X3D/VRML. A browser plug-in at the client side makes it possible to view the VRML or X3D output. VRML and X3D provide the possibility to start a script when a user clicks on an object. This functionality is used to retrieve the non-spatial information that is linked to a 3D geo-object. Via the VRML/X3D plug-in a request is sent to a (application) server. The server receives and interprets the incoming information and sends a HTML with the required information back to the browser.

For retrieving the spatial and the non-spatial information from the DBMS a technique is needed to communicate between a client and a database on a server. For this communication, several techniques are available such as ColdFusion, ASP.NET, ASP, JSP or PHP. The choice of the used technique is dependent on the used web server.

The detailed architecture of publishing a 3D dataset on the web is shown in Figure 3.12 below:

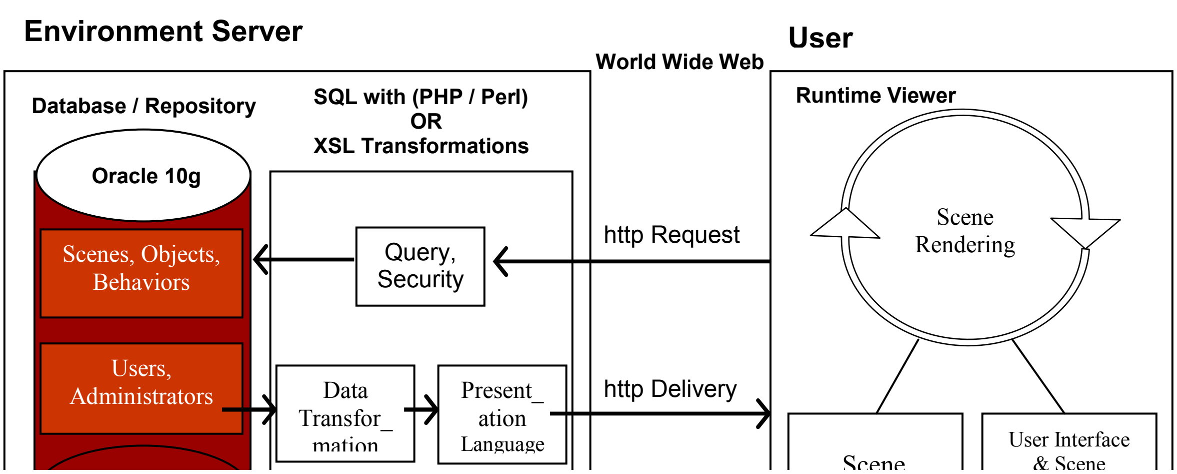


Figure 3.12: Web Publishing Architecture of 3D Data Sets Using 3D-GeoDBMS

To show the possibilities to query 3D geo-objects via an Internet client, first a simple prototype was built, based on Microsoft technology.

4.6 Summary

3D spatial data modeling is one of the key research problems in 3D GIS. More and more applications depend on these 3D spatial data. Mostly, these data are stored in Geo-DBMSs. However, recent Geo-DBMSs do not support 3D primitives modeling, it only able to describe a single-attribute of the third-dimension, i.e. modeling 2.5D datasets that used 2D primitives (plus a single z-coordinate) such as polygons in 3D space. This chapter focuses on 3D topological model based on space partition for 3D GIS, for instance, 3D polygons or tetrahedron form a solid3D object. Firstly, this paper discusses formal definitions of 3D spatial objects, and then all the properties of each object primitives will be elaborated in detailed. It also discusses methods for constructing the topological properties to support object semantics is introduced. The formal framework to describe the spatial model, database using Oracle Spatial is also given in this chapter. All related topological structures that forms the object features are discussed in detail. All related features will be tested using real 3D spatial dataset of 3D building given in next chapter.

CHAPTER 5

EXPERIMENTS AND RESULTS

5.1 Introduction

In this chapter, model testing is involved the implementation of the model using an object-relational database management system. As a part of the test implementation, a data set of strata plans (architecture drawing) of an area closed UTM campus (Skudai) is utilized. This chapter summarizes and discusses the performance test results of the research.
A number of interesting issues are presented. The discussion of the proposed a model is also presented.

5.2 Oracle Spatial Geometry Schema

In order to visualize geometrical objects, geometry column for each of the spatial primitives, i.e. node, line, face, and solid are created. The purpose of preparing these geometry columns is to integrate the spatial database into Map 3D for the purpose of displaying object graphics.

A geometry column consists of geometry object that represent a spatial feature, modeled as an ordered set of primitive elements, i.e. node, line, and polygon. In Oracle Spatial, the geometry column consists of a schema (MDSYS) that prescribes the storage, syntax, and semantics of supported geometric data types. The pre-defined object type SDO_GEOMETRY is:

CREATE TYPE sdo_geometry AS OBJECT (SDO_GTYPE NUMBER,
SDO_SRID NUMBER, SDO_POINT SDO_POINT_TYPE,
SDO_ELEM_INFO MDSYS.SDO_ELEM_INFO_ARRAY
(SDO_STARTING_OFFSET, SDO_ETYPE, SDO_INTERPRETATION),
SDO_ORDINATES MDSYS.SDO_ORDINATE_ARRAY)
(list of ordinates);

Note: Not In List (NIL)

For example (see Figure 5.1):

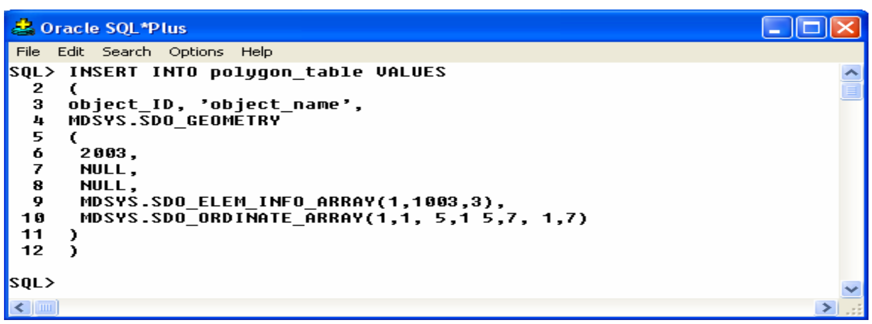


Figure 5.1: Data set insertion into Polygon table

5.3 AutoDesk Map 3D Schema Table

In the AutoDesk Map 3D 2005 (Map 3D), there is a pre-defined Oracle schema specification that need to be fulfilled. The imported spatial database for visualization is highly depends on this schema. One requires to create the same Map 3D's schema as;

(i) Primary schema table, and

(ii) Secondary tables' columns schema.

In primary schema described in Table 5.1, tables need to be created as follows:

Table 5.2: Autodesk Map Schema in Oracle 10g

TAB_NAME	**TAB_TYPE**	
ADMPOSEMETADATA	TABLE	
ADMPMETADATA	TABLE	
ADMPFEATURELAYER	TABLE	
ADMPOPTIONS	TABLE	
FACE	*TABLE*	--user-defined table
LINE	*TABLE*	--user-defined table
NODE	*TABLE*	--user-defined table
SOLID	*TABLE*	--user-defined table
ADMPIMPORTSETTINGS	TABLE	

After the primary schema tables are created, each of these tables are filled with specific columns that the Map 3D requires.

In order to create user-defined spatial and topological database, i.e. node, line, face, and solid tables, we have to add two additional tables such as Geometry and EntityID which includes the geometrical data and the specific id for any entity respectively.

5.4 Synchronization between Oracle and Map 3D schemas

In order to visualize Oracle spatial database successfully within the Map 3D environment, synchronization between both Oracle spatial and Map 3D schema is required. However, spatial datasets are inserted into Oracle Spatial as follows:

Sample spatial datasets:

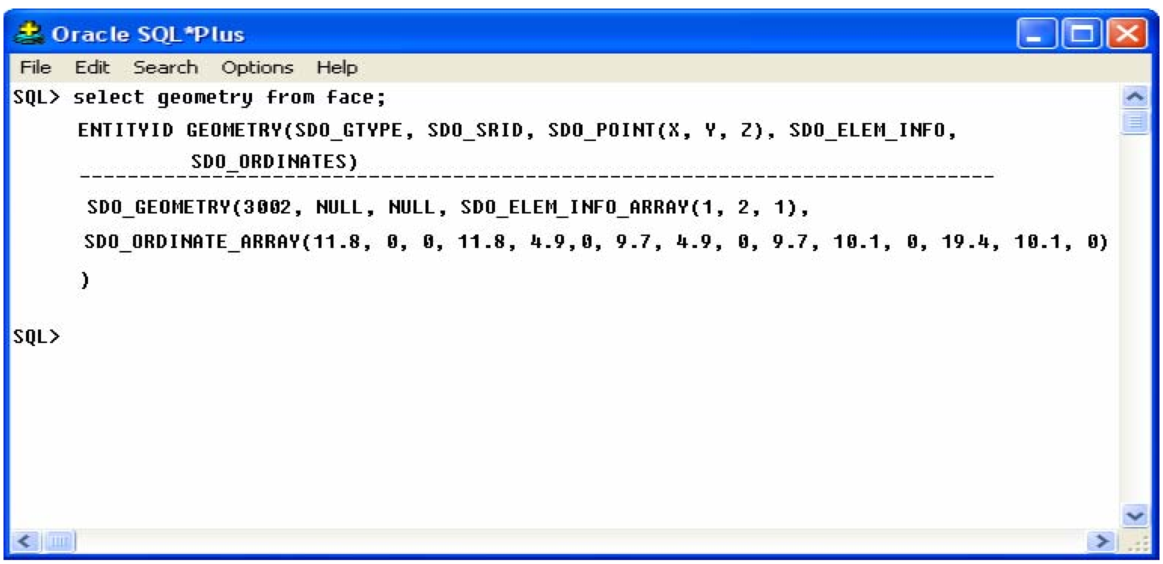


Figure 5.2: Extraction of geometry dataset

After datasets are inserted into Oracle database, the first stage of synchronization is to login into Map 3D. Later on, the second stage of synchronization is to connect Oracle schema table into Map 3D's schema administration. The Map 3D will prompt user for Oracle database login name and password. From the Oracle database lists, select the appropriate database for visualization.

After the synchronization was included into Map 3D, databases are imported from Oracle schema table to Map 3D. Features selection must be identified in order to display the appropriate dataset. Figure 5.3 denotes the methodology to extract dataset from Oracle database to Map 3D, where Figure 5.4 display the Oracle dataset within Map 3D environment.

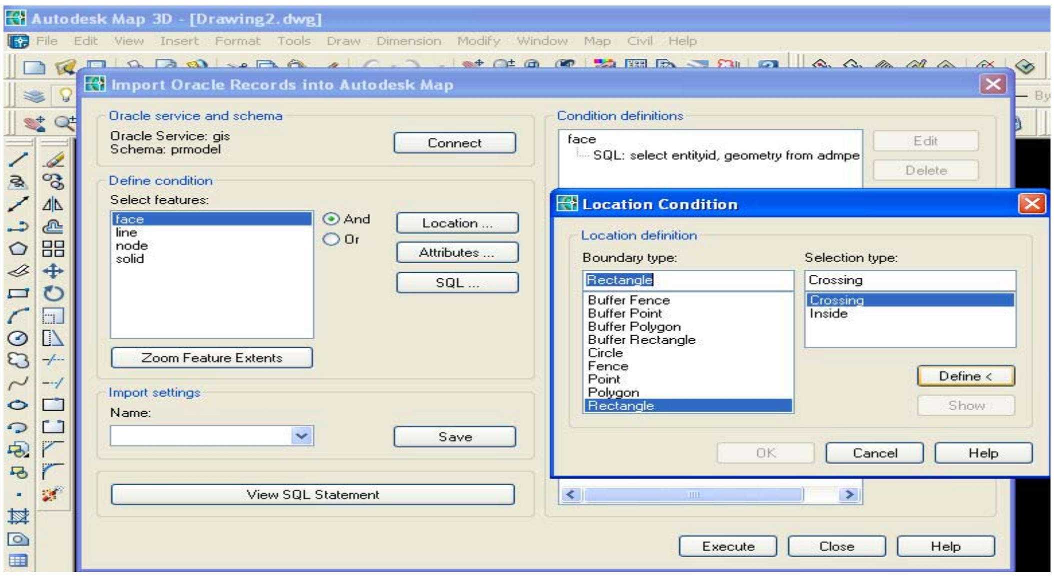


Figure 5.3: Import Oracle Records into Map 3D

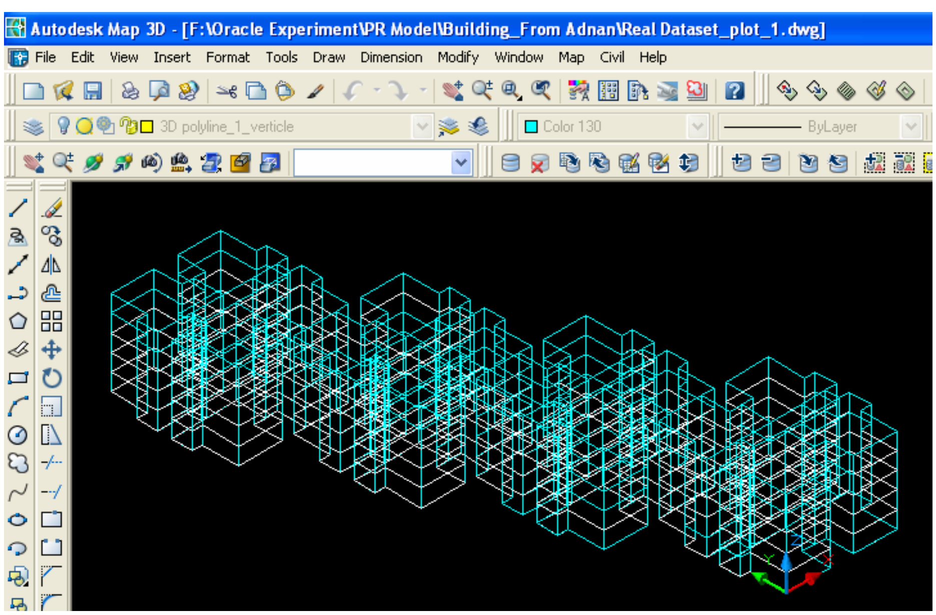
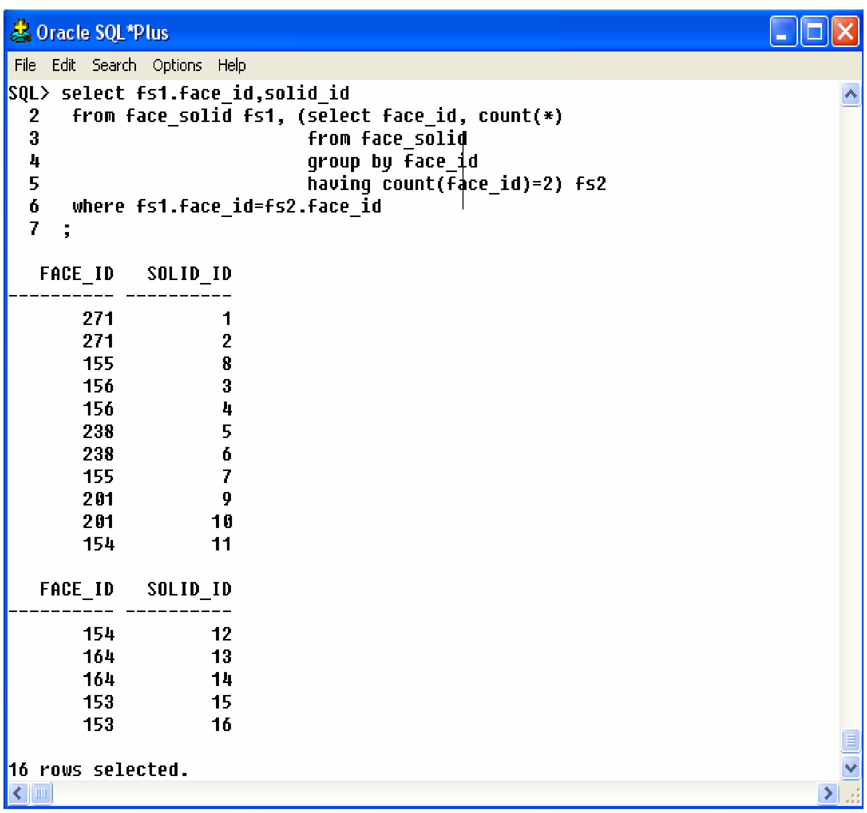


Figure 5.4: Final Oracle datasets display

5.5 Querying 3-D Condense Spatial (CoS) Model

Autodesk Map 3D is a tool for visualizing spatial data managed by Oracle Spatial. Autodesk Map 3D and Oracle Spatial combined can also serve as a powerful GIS platform, in that it not only to directly work on a modern spatial database, but also be able to immediately visualize the work they have done in the database and also can perform the analysis.

For instance, the following query shows the results of common walls between different apartments in a same building. Following screenshot in Figure 5.5 displays a SQL query operation and its textual result, followed by a screenshot in Figure 5.6 showing the same query result displayed through Autodesk Map 3D.



Oracle SQL*Plus

File Edit Search Options Help

```
SQL> select fs1.face_id,solid_id
  2   from face_solid fs1, (select face_id, count(*)
  3                         from face_solid
  4                         group by face_id
  5                         having count(face_id)=2) fs2
  6   where fs1.face_id=fs2.face_id
  7  ;

   FACE_ID   SOLID_ID
---------- ----------
       271          1
       271          2
       155          8
       156          3
       156          4
       238          5
       238          6
       155          7
       201          9
       201         10
       154         11

   FACE_ID   SOLID_ID
---------- ----------
       154         12
       164         13
       164         14
       153         15
       153         16

16 rows selected.
```

Figure 5.5: Nested SQL query, showing common faces in 3D Solids

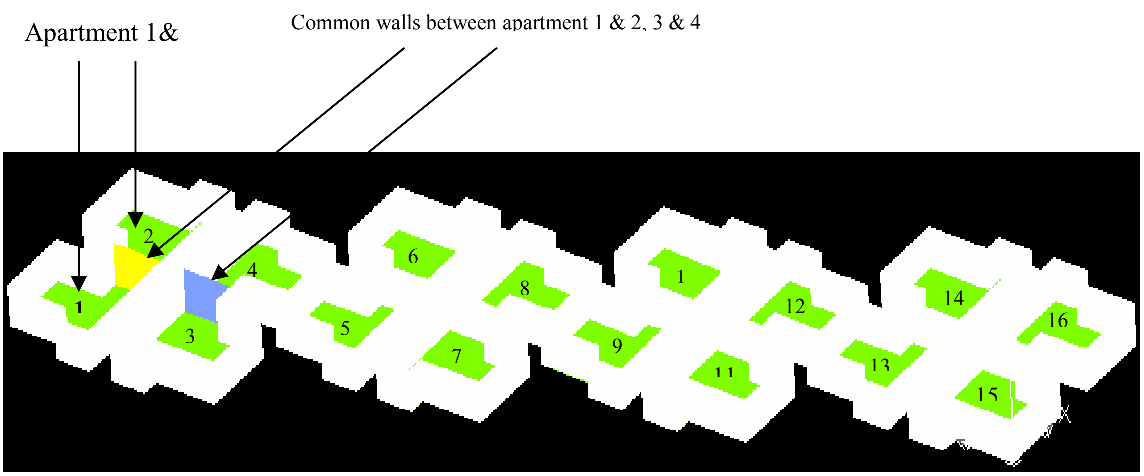


Figure 5.6: Nested SQL query, showing common walls in 3D apartments

The screenshot in Figure 5.6 shows complex 3D apartments on ground floor of a building which is generated by Autodesk Map 3D. Note this experimental application we built is actually also utilizing the geo-coding capabilities of Oracle Spatial. In this particular project 3D data of all the floors of building is loaded into Oracle Spatial.

5.6 Dynamic 3D Scene Generation on Web

Today, the dynamic generation of HTML pages is a standard functionality of all commercial database systems. This feature has been proven to be a very effective and practical approach to support the (two-dimensional) visualization of information stored within database systems. In this research, we outline how a similar functionality can be realized to dynamically generate VRML scenes from a database management system (DBMS). This approach overcomes many of the limitations of static VRML scenes, by exploiting the persistence, scalability and security mechanisms of database management systems (Codd, 1970). In addition, it also provides a direct way to efficiently generate three dimensional visualizations from existing information in the database.

In the experiment shown here, a "Geometry" data type allows, for example, to store all apartments of a building as VRML scenes together with the walls and its floor information along with the its ID. It is possible to select a subset of all apartments with certain properties and merge them in a new scene, e.g., in order to display all apartments that have already been rented to customers.

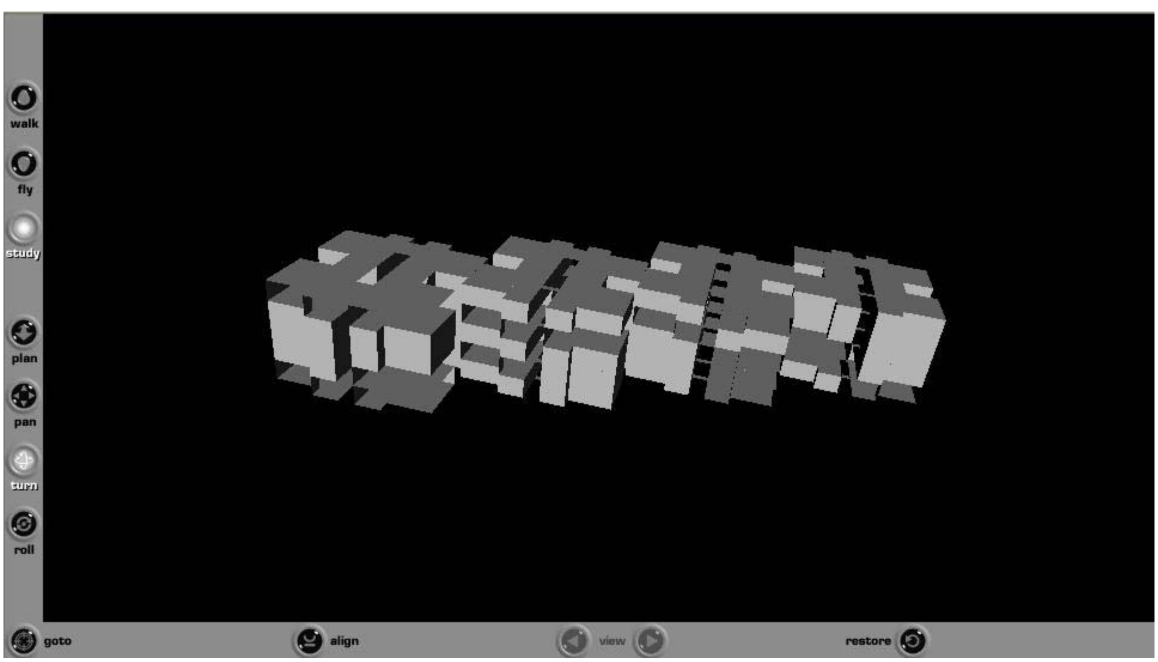


Figure 5.7: 3D dynamic data visualization of 3D building data on Web

In the above Figure 5.7, a complete building structure is shown using the VRML client application installed as a plug-in with web browser. As it is shown in the above Figure 5.6 that how complex 3D data is and to generate the dynamic scene through database onto the web browser was using the detailed architecture as shown above in the Figure 5.5.

Dynamic Scene generation in VRML/X3D using the data from the Oracle Spatial 10g has done in a way given below:

A database connection is used as a statement inside PHP coding. Here are the strings used to connect to the Oracle database:

```
<?
$connection = OCILogon ("User Name", "Password", "service name")
or die ("cannot connect to database");
?>
```

The SQL in this case is quite simple—return all data contained in the table after execution:

```
<?
   $stmt = OCIParse($connection, "select * from building" )
              or die ("cannot select");
   OCIExecute($stmt, OCI_DEFAULT);
?>
```

Once the result is assembled, the code loops through all of the records and displays as many Floors as there are records in the database. This is the loop:

```
<?
$count = 0;
While (OCIFetch ($stmt)) {
color = ociresult ($s, "color");
radius = ociresult ($s, "radius");
?>

DEF Floor <? $count ?> Transform {
translation <? $count*15 ?> 0 0
children [
Shape {
appearance Appearance {
material Material {
diffuseColor <? $color ?>}
}
geometry Sphere {
radius <?$radius?>
} } ] }
<?
count++;
}
```

Notice how the values from the database are inserted into the VRML. Two strings, color and radius, are set to the values from the database and then displayed in the VRML code as <? $color ?> and <? $radius ?>.

The translation is handled by an integer (count), which keeps track of the number of records and displaces the floor by 15 on the x-axis every iteration of the loop.

The variable count has another use. As we generate objects, we give them a name: DEF Floor1, DEF Floor2, and so on. We do this by inserting the count into the VRML node's definition DEF Floor <? $count ?>

A DBMS supporting three dimensional visualization must be able to generate new VRML scenes, both from existing operational business data represented by conventional data types and from existing multimedia data represented by specialized media data types. Within the Building, statistical data about how many persons can live in one apartment could be visualized by an arrow diagram, where the arrow size is proportional to the number of family members allowed to live in an apartment. As always, free the resources database resources at the end.

```
<?
   OCIFreeStatement ($stmt);
    OCILogoff ($connection);
?>
```

The VRML client has to be able to directly read and write the DBMS from within a VRML scene. For more advanced interaction modes in multi-user environments, this mechanism needs to be complemented by an event handling system. This allows signaling a change in a scene to all other users actively working on the same scene. The VRML event handling of the other users can then react by appropriately updating the scene, e.g. by reloading a part or the whole scene.

The most important benefit of the above-outlined approach is that by means of storing VRML scenes within a DBMS, we achieve persistence of changes to scenes. Furthermore, the multi-user access control enables the sharing of VRML data among multiple users, thus we move from isolated, static scenes to shared spaces of dynamically generated three-dimensionally visualized information. Scalability is achieved by loading and generating scenes and scene components dynamically either at loading time or at run time. The corresponding loading/generation schemes can be determined both by physical characteristics of the VRML scene and the logical structure of the application. Controlled access to scene data is supported by the security and view mechanisms of the underlying DBMS. For example, in the "Building" scenario, one might display the rent of an apartment only to authorized members or customers but not to visitors.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 Conclusion and Future Work

Spatial data management using Geo-DBMS become one of the leading technologies recently. Although its 3D spatial data capability currently does not offer wider spectrum of analytical tools, its overall flexibility enables user to implement at least some of it with little effort, i.e. spatial queries, and dataset retrieval. Spatial data types for 2D are currently available in mainstream DBMSs and now geo-science is focusing on support for 3D, 4D and topology in DBMSs. The next step is to make the data organized in a geo-DBMS available within and between organizations and for individual users via Web technology.

This thesis contributes to these developments, since it illustrates how 3D visualization techniques and techniques to query DBMSs via a web server can be combined. We showed how 3D geometry stored in an Oracle database can be converted into VRML or X3D, and how the 3D objects can be presented in a 'simple' Internet browser together with their non-spatial attribute information. Future work will focus on a number of research issues.

The described data model presented in this thesis could make a basic development of facilities management system of almost any scale, i.e. building management, underground construction, or 3D strata title databasing; enabling user to perform basic spatial, attribute and temporal analysis (or any combination of these) without the need to extensively further develop it. Its topologically structured spatial component ensures high data consistency with low redundancy.

Further research should be oriented to extending 3D spatial analytical capabilities either by adopting new native functionality such as spatial operators, or by user programmed extensions. It should also be tested which of current 3D data modeling standards like GML and X3D suits better for similar systems needs. Both technologies enable combination of spatial and non- spatial data enabling system a better client-server load balancing which is at this point mostly sever oriented and requires a round trip for most of additional queries.

We have discussed some of the VRMLX3D and Geo-DBMS's features which we used for animation and user interaction with the rendered 3D model. With our approach, we have achieved the most of the requirements for dynamic information visualization support we have identified. Security and persistence are provided as a basic property of the DBMS.

Although experience with the prototype looks promising with respect to performance, serious tests on larger data sets will be set up. Fast rendering of 3D objects is of course critical when displaying data via the Internet. One option is the use of compression techniques. Other strategies have to do with navigation and zoom-levels. First data with low Level of Detail can be presented, and only when the user zooms in the Level of Detail is increased (Kofler, 1998). This touches the fundamental research issue of storing multi-representations in DBMS's or performing generalizations on the fly.

Another issue is that of visualization constraints. X3D and VRML environments were originally meant to visualize all kinds of objects. Therefore, constraints should be implemented in the prototypes to meet specific conditions for visualizing geo-information, e.g. one should not be able to turn the data set upside down.

The prototype described in this thesis form a contribution to accessing 3D geo-objects organized in a DBMS. Our experiences show potentials for the use of common Web technology in GIS and more specific in 3D GIS applications, which should be further elaborated in future research.

References and Bibliography

Abdul-Rahman, A. (2000). *The Design and Implementation of a Two and Three-Dimensional Triangular Irregular Network Based GIS.* PhD Thesis, University of Glasgow.

Arens, C., Stoter J., and Oosterom P. v (2003). *Modeling 3D spatial objects in a GEO-dbms using a 3d primitive.* Proceedings of the 6th AGILE Conference, Lyon, France. (URL: http://www.gdmc.nl/stoter/publicaties/agile2003.zip).

Albert, J., Bachmann, M., Hellmaier, A. (2003). Zielgruppen und Anwendungen für Digitale Stadtmodelle und Digitale Geländemodelle. (URL: http://www.ikg.uni-bonn.de/sig3d/docs/Tabelle_Anwendungen%2BZielgruppen.pdf)

Altmaier, A., and Kolbe, T. (2003). *Applications and Solutions for Interoperable 3d Geo-visualization.* Proceedings of the Photogrammetric Week 2003, Stuttgart, Germany. (URL: http://www.ikg.uni-bonn.de/sig3d/docs/Altmaier_und_Kolbe_PhoWo2003_englisch.pdf)

Billen, R., Zlatanova, S., Mathonet, P., and Bonvier, F. (2002). *The Dimensional Model: a framework to distinguish spatial relationships.* Advances in Spatial Data Handling, 10th International Symposium on Spatial Data Handling – Ottawa, D.Richardson and P.van Oosterom (Eds.), pp. 285-298. Springer-Verlag, Berlin.

Cöltekin, A. (2002). *An Analysis of VRML-based 3D Interfaces for Online GIS: Current Limitations and Solutions.* Surveying Science in Finland, Vol.20, No: 1-2, pp.80-91. (URL: http://foto.hut.fi/~arzu/publications/ColtekinA_SSF2002.pdf).

Connolly, T., and Begg, C. (2002). *Database Systems – A Practical Approach To Design, Implementation and Management.* 3rd Edition, Pearson Addison-Wesley, Menlo Park, California.

Coors, V., and Flick, S. (1998). *Integrating Levels of Detail in a Web-based 3D-GIS.* Proceedings of the 6th ACM international symposium on advances in geographic information systems, Washington D.C., USA. (URL: http://doi.acm.org/10.1145/288692.288701).

Coors, V., and Jung, V. (1998). *Using VRML as an Interface to the 3D Data Warehouse.* 3rd Symposium on the Virtual Reality Modeling Language (VRML98), Monterey, CA, USA, Feb. 16-19. (URL: http://doi.acm.org/10.1145/271897.279553).

Coors, V. (2003). *3D GIS in Networking Environments.* Computers, Environment and Urban Planning 27/4 (2003), Special Issue 3D cadastre, pp 345-357. (URL: http://dx.doi.org/10.1016/S0198-9715(02)00035-2).

Coyle, F. (2002). *XML, Web Services and the Changing Face of Distributed Computing.* (URL: http://www.acm.org/ubiquity/views/f_coyle_1.html).

Diehl, S. (2001). *Distributed Virtual Worlds – Foundations and Implementation Techniques Using VRML, JAVA, and CORBA.* Springer Verlag, Berlin Heidelberg New Work.

Dodge, M., Doyle, S., Smith A., and Fleetwood S. (1998). *Towards the virtual city: VR&Internet GIS for urban planning.* Virtual Reality and Geographical Information Systems Workshop, 22nd May 1998, Birkbeck College, London, 12 p.

Döllner, J. (2002). *3D-Visualisierung.* Proceeding of Webmapping 2002 – Workshop.

Doyle, A., and Cuthbert, A. (1998). *Essential Model of Interactive Portrayal.* OpenGIS project document 98-061.

Dykes, J.A., Moore, K.E., and Fairbairn D. (1999). *From Chernoff to Imhof and Beyond: VRML and Cartography.* Proceedings of 4th Int. Conference on the VRML and Web3D Technologies (VRML99), Paderborn, Germany.

ebXML (2004). *Enabling a Global Electronic Marketplace.* (URL: http://www.ebxml.org/).

Egenhofer, M. (1994). *Spatial SQL: A Query and Presentation Language.* IEEE Transactions on Knowledge and Data Engineering, 6 (1), pp.86-95. (URL: http://www.spatial.maine.edu/~max/SSQL.pdf).

Egenhofer, M., and Herring, J.R. (1991). *High-Level Spatial Data Structures for GIS.* Geographical Information Systems, Vol. 1: Principles, D. Maguire, M. Goodchild, and D. Rhind (eds.), Longman, London, pp. 227-237, 1991.

Erlanger, L. (2002). *Distributed Computing: An Introduction.* (URL: http://www.extremetech.com/print_article/0,3428,a=25002,00.asp).

ESRI (2001). An *Overview of the Geography Network – An ESRI White Paper.* (URL: http://www.esri.com/library/whitepapers/pdfs/gn_overview.pdf).

Franklin, S. (2001). *Object Oriented Databases are worth a closer look.* (URL: http://archive.devx.com/dbzone/articles/sf0601/sf0601p.asp).

GeoVRML (2004). *GeoVRML – A Web3D Working Group.* (URL: http://www.geovrml.org/).

Gisolfi, D. (2001). *Web Service architect, Part 3: Is Web services the reincarnation of CORBA?.* IBM, developerWorks, Web services, Web services articles (URL: ftp://www6.software.ibm.com/software/developer/library/ws-arc3.pdf).

Gröger, G., Kolbe, T., Dress, R., Müller, H., Knopse, F., Gruber, U., and Krause, U. (2004). *Das interoperable 3D-Stadtmodell der SIG 3D der GDI NRW.* Version 2. Stand: 10.5.2004 (URL: http://www.ikg.uni-bonn.de/sig3d/docs/Handout_04_05_10.pdf).

Gruber, M. (1999). *Managing Large Urban Databases.* Photogrammetric Week, Heidelberg, 1999, pp.341-349.

Güting, H. (1994). *An Introduction to Spatial Database Systems.* VLDB Journal 3, Hans-J. Schenk (Editor), pp.357-399, 1994.

Haber, R.B., and McNabb, D.A. (1990). *Visualization Idoms. A Conceptual Model for Scientific Visualization Systems.* (URL: http://www.siggraph.org/education/materials/HyperVis/concepts/idio_sh.htm).

ISO/TC 211 (2004). (URL: http://www.isotc211.org/).

Jansen, R. (2003): *Oracle, Java, XML – Integration in Oralce 9i*. Software & Support Verlag.

Kolodziej, K. (2003). *OpenGIS Web Map Server Cookbook.* OpenGIS Consortium (URL: http://www.ogcnetwork.org/docs/03-050r1.pdf).

Kofler, M. (1998). *R-Trees for Visualizing and Organizing Large 3D Databases.* Dissertation zur Erlangung des akademischen Grades Doktor der Wissenschaften, TU Graz.

Kwan, M.-P., and Lee, J. (2003). *Emergency response after 9/11: the potential of real-time 3D GIS for quick emergency response in micro-spatial environments.* In Press, Corrected Proof, Available online 28 September 2003. (URL: http://dx.doi.org/10.1016/j.compenvurbsys.2003.08.002).

Lake, R. (2004). *The Future of Interoperability.* GEO World, May 2004. (URL: http://www.geoplace.com/gw/2004/0405/0405int.asp).

Laurini, R. (2001). *Information System For Urban Planning – A hypermedia co-operative approach.* London New York.

Lopez, X. (2003). *Oracle Database 10g: A Spatial VLDB Case Study.* Oracle Cooperation Whitepaper. (URL: http://otn.oracle.com/products/spatial/pdf/customer_success/papers/spatial_10g_ow2003.pdf).

Molenaar, M. (1998). *An Introduction to the Theory of Spatial Object Modelling for GIS.* Taylor & Francis London Bristol.

Molenaar, M. (1990). *A formal data structure for 3D vector maps.* Proceedings 1st European Conference on Geographical Information Systems (EGIS'90), Volume 2, Amsterdam, The Netherlands, pp.770-781.

Maruyama, H. *et al* (2002): *XML and Java – Developing Web Applications.* Second Edition. Boston San Francisco New York.

Microsoft Corp. (2004). *Microsoft MapPoint Web Service: the power of location – any place, any time, any device.* (URL: http://www.microsoft.com/mappoint/webservice/default.mspx).

Nagappan, R., Skoczylas, R, and Sriganesh, R.P. (2003). *Developing Java Web Services – Architecting and Developing Secure Web Services Using Java.* Indianapolis, USA.

Nebiker, S. (2003). *Support For Visualization And Animation In A Scalable 3D GIS Environment: Motivation, Concepts and Implementation.* ISPRS Commission V, WG V/6, International Workshop on Visualization and Animation of Reality-based 3D Models, Vulpera (Engadin), Switzerland (= International Archives of Photogrammetry, Remote Sensing and Spatial Information Science, Vol. XXXIV-5/W10).

Nedovic-Budic, Z., Feeney, M.-E., Rajabifard, A., and Williamson, I. (2003). *Are SDIs serving the needs of local planning? Case study of Victoria, Australia and Illinois, USA.* Computers, Environment and Urban Systems, Volume 28, pp.329-351.

Oestensen, O. (2001). *The expanding agenda of Geographic information standards.* (URL: http://www.iso.org/iso/en/commcentre/pdf/geographic0107.pdf).

Open GIS Consortium (2004a). *OGC Web Services, Phase 2 (OWS-2) - An OpenGIS Initiative.* (URL: http://www.opengis.org/initiatives/?iid=7).

Open GIS Consortium (2004b). *The Spatial Web – An Open GIS Consortium (OGC) White Paper.* (URL: http://portal.opengis.org/files/?artifact_id=3859&version=1&format=pdf).

Open GIS Consortium (2003). *OpenGIS Reference Model. Version 0.1.2*. (URL: http://www.opengis.org/docs/03-040.pdf).

OpenGIS Consortium (2001). *The OpenGIS Consortium Abstract Specification – Topic 1: Feature Geometry.* (URL: http://www.opengis.org/docs/01-101.pdf)

Open GIS Consortium (1999). *OpenGIS Abstract Specification – Topic 0: Abstract Specification Overview.* (URL: http://www.opengis.org/docs/99-100r1.pdf).

Oosterom, P. v, Stoter, J., Quak, W., and Zlatanova, S. (2002). *The Balance Between Geometry and Topology.* Advances in Spatial Data Handling, 10th International Symposium on Spatial Data Handling - Ottawa, D.Richardson and P.van Oosterom (Eds.), Springer-Verlag, Berlin, pp. 209-224.

Peng, Y.-R., Tsou, M.-H. (2003). *Internet GIS – Distributed Geographic Information Services for the Internet and Wireless Networks.* Hoboken, New Jersey, USA.

Pfund, M. (2001). *Topologic Data Structure For a 3D GIS.* Proceedings of the 3rd ISRS Workshop on Dynamic and Multidimensional GIS, pp.233-237, Bangkok, Thailand.

Pigot, S. (1992). *A Topological Model for a 3D Spatial Information System.* Proceedings of 5th Symposium on SDH, Vol.1, Charleston, USA, pp.344-359.

Pilouk, M. (1996). *Integrated Modeling fors 3D GIS.* PhD Thesis, Wageningen Agricultural University and ITC Enschede.

Ramsey, P. (2003). *PostGIS Manual.* (URL: http://postgis.refractions.net/docs/postgis.pdf).

Ravada, S. (2003). *Topology Management in Oracle 10g.* Proceedings Dagstuhl Seminar 03041, Sept 2003, Germany. (URL: ftp://ftp.dagstuhl.de/pub/Proceedings/03/03401/03401.RavadaSiva.Slides.ppt).

Reddy, M., Iverson, L., Leclerc, Y., and Heller, A. (2001). *GeoVRML: Open Web-based 3D Cartography.* Proceedings of the International Cartographic Conference (ICC2001), Beijing, 6-10 August 2001 (URL: http://www.ai.sri.com/~reddy/pubs/pdf/icc2001_geovrml.pdf).

Reichardt, M. (2003). *The Havoc of Non-Interoperability.* OGC White Paper (URL: http://portal.opengis.org/files/files/?artifact_id=5097).

Rigaux, P., Scholl, M., and Voisard, A. (2002). *Spatial Databases – with Application to GIS.* San Francisco.

Rosa, A. de la, and Cervelle B. (1999). *3D Topological modeling and visualisation for 3D GIS.* Computers & Graphics, Vol. 23, Issue 4, pp.469-478.

Savarese, D.F. (2003). *Learning to Fly.* Java Pro Magazine, June issue (URL: http://www.fawcette.com/javapro/2003_06/magazine/features/dsavarese/).

Shekar, S., and Chawla, S. (2003): *Spatial Databases – A Tour.* Pearson Education, New Jersey.

Shi, W., Yang, B., and Li, Q. (2003). *An object orientated data model for complex objects in three-dimensional geographical information systems.* International Journal of Geographical Information Science, Vol. 17, No.5, pp. 411-430.

Special Interest Group (SIG) 3D (2004). *Pilot 3D der GDI NRW – Ergebnisse.* (URL: http://www.ikg.uni-bonn.de/sig3d/docs/040109_Flyer_Endergebnis_3D-Pilot.pdf).

Stoter, J., and Oosterom, P. v (2002). *Incorporating 3D Geo-Objects into a 3D Geo-DBMS.* Proceedings ACSM-ASPRS 2002. (URL: http://www.gdmc.nl/stoter/publicaties/ASPRS2002.zip).

Stoter, J., and Zlatanova, S. (2003a). *3D GIS, where are we standing?.* Joint Workshop on Spatial, Temporal and Multi-Dimensional Data Modeling and Analysis, Quebec City, Canada (URL: http://www.gdmc.nl/zlatanova/thesis/html/refer/ps/st_zl_03.pdf).

Stoter, J., and Zlatanova, S. (2003b). *Visualisation and editing of 3d objects organised in a DBMS.* Proceedings of the EuroSDR Com V. Workshop on Visualisation and Rendering, 22-24 January 2003, Enschede, The Netherlands.

Sun Microsystems (2004): *The Java 3D API.* (http://java.sun.com/products/java-media/3D/).

Tempfli, K., and Pilouk, M. (1994). *Vector Data Models for 3D GIS.* Proceedings of the 3rd International Colloquium of LIESMARS on integration, automation and intelligence in Photogrammetry, RS and GIS, Wuhan, China, pp.272-283.

Tsichritzis, D., and Klug, A. (1978). *The ANSI/X3/SPARC DBMS framework - Report of the study group on database management systems.* Information Systems, vol. 3, pp.173-191.

Vasudevan, V. (2001). *A Web Service Primer.* (URL: http://webservices.xml.com/pub/a/ws/2001/04/04/webservices/index.html).

Vries, M.E. de, and Stoter J. (2003). *Accessing 3d geo-DBMS using Web technology.* ISPRS Joint Workshop on "Spatial, Temporal and Multi-Dimensional Data Modelling and Analysis", Quebec, Canada, October 2003.

Web3D Consortium (2004). (URL: http://www.web3d.org/).

X3D GeoSpatial (2004). *X3D GeoSpatial Working Group.* (URL: http://www.web3d.org/x3d/workgroups/geospatial.html).

Zhu, C., Tan, E.C., and Chan, K.Y. (???). *3D Terrain visualization for Web GIS.* (URL: http://www.gisdevelopment.net/technology/ip/pdf/ma03065.pdf).

Zlatanova, S. (2000). *3D GIS in Urban Development.* PhD Thesis, Graz Technical University. (URL: http://www.gdmc.nl/zlatanova/PhDthesis/pdf/content.html).

Zlatanova, S., and Holweg, D. (2004). *3D Geo-Information in Emergency Response: A Framework.* Proceedings of the 4th Int. Symp. on Mobile Mapping Technology (MMT'2004), March 29-31, Kunming, China (URL: http://www.gdmc.nl/zlatanova/thesis/html/refer/ps/sz_dh_mmt.pdf).

Zlatanova, S., Hohlweg, D., and Coors, V. (2003). *Geometrical and Topological Models for Real-Time GIS.* Proceedings of the Next Generation Geospatial Information Workshop 2003, Cambridge, Massachusetts, USA.

Zlatanova, S., Abdul-Rahman, A., and Shi, W. (2004). *Topological models and frameworks for 3D spatial objects*. Journal of Computers & Geosciences, May, Vol 30, Issue 4, pp. 419-428.

Zlatanova, S., Abdul-Rahman, A., and Pilouk, M. (2002). *3D GIS: Current Status and Perspectives.* Proceedings of the Joint Conference on Geo-spatial theory, Processing and Applications, 8-12 July, Ottawa, Canada. (URL: http://www.geo.tudelft.nl/frs/papers/2002/Sisi%203DGIS%20Ottowa.pdf).